Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Навчально-науковий
інститут електроенергетики
(навчально-науковий інститут)

Факультет інформаційних технологій
(факультет)

Кафедра інформаційних технологій та комп'ютерної інженерії
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня магістра

Здобувача вищої освіти Ширмакова Леоніда Олеговича
(ПІБ)

академічної групи 123-23м-1
(шифр)

спеціальності 123 Комп'ютерна інженерія
(код і назва спеціальності)

за освітньо-професійною програмою «Комп'ютерна інженерія»
(офіційна назва)

на тему «Обгрунтування параметрів мережевих пристроїв комп'ютерної системи ДП «Антонов» з врахуванням впровадження системи безпеки відеонагляду та IP телефонії»
(назва за наказом ректора)

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
Кваліфікаційної роботи:	доц. Шедловський І.А.			
Розділів:				
Синтез системи	доц. Ткаченко С.М.			
Розробка програмного забезпечення	ас. Бешта Л.В.			

Рецензент				

Нормоконтролер	проф. Цвіркун Л.І.			

Дніпро
2024

ЗАТВЕРДЖЕНО:
завідувач кафедри
інформаційних технологій
та комп'ютерної інженерії
(повна назва)

_____ В.В. Гнатушенко
(підпис) (ініціали, прізвище)

« » _____ 2024 року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня магістра
(бакалавра, магістра)

здобувача вищої освіти Ширмакова Л.О **академічної групи** 123-23м-1
(прізвище та ініціали) (шифр)

спеціальності 123 Комп'ютерна інженерія

за освітньо-професійною програмою «Комп'ютерна інженерія»
(офіційна назва)

на тему «Обґрунтування параметрів мережевих пристроїв комп'ютерної системи ДП «Антонов» з врахуванням впровадження системи безпеки відеонагляду та IP телефонії»

затверджену наказом ректора НТУ «Дніпровська політехніка» від 17 жовтня 2024 № 1388-с

Розділ	Зміст	Термін виконання
Стан питання та постановка завдання	На основі вихідних даних практик та відповідних науково-технічних джерел сформулювати наукове завдання, визначити предмет і мету дослідження.	15.10.2024
Теоретичний розділ	Дослідити теоретичні основи побудови комп'ютерної мережі підприємства. Визначити методи оцінки роботи пристроїв, методику метрик мережі.	26.10.2024
Синтез системи	Побудувати систему підприємства, розрахувати параметри мережі. Провести вибір засобів для забезпечення ефективності мережі.	10.11.2024
Розроблення програмного забезпечення	Розробити програмне забезпечення для генерації трафіку та оцінки мережевих параметрів у рамках дослідження	20.11.2024
Експериментальний розділ	Побудувати комбіновану модель мережі. Провести експеримент використовуючи математичну та комбіновані моделі	05.12.2024

Завдання видано _____ доц. І.А. Шедловський
(підпис керівника) (ініціали, прізвище)

Дата видачі 06 вересня 2024

Дата подання до екзаменаційної комісії 10.12.2024

Прийнято до виконання _____ Л.О. Ширмаков
(підпис здобувача вищої освіти) (ініціали, прізвище)

РЕФЕРАТ

Пояснювальна записка: 133 с., 43 рис., 10 табл., 1 дод., 30 джерел.

КОМП'ЮТЕРНА СИСТЕМА, МОДЕЛЬ, IP-ТЕЛЕФОНІЯ, СИСТЕМА ВІДЕОНАГЛЯДУ, МЕРЕЖЕВЕВИЙ ПАКЕТ, АНАЛІЗ ТРАФІКУ.

Об’єктом дослідження є комп’ютерна система ДП «Антонов», яка включає в себе інфраструктуру для відеонагляду та IP-телефонії. Метою роботи є обґрунтування параметрів мережевих пристроїв цієї системи, з урахуванням специфіки впровадження системи безпеки відеонагляду та IP-телефонії. Робота спрямована на оптимізацію роботи мережі, підвищення її ефективності, а також забезпечення безперебійної роботи в умовах навантажень та можливих атак.

Для досягнення мети були використані методи моделювання мережі в середовищі EVE-NG, а також реальне апаратне забезпечення, включаючи сервери, телефонний шлюз та комутатор Cisco. Це дозволило створити комбіновану модель мережі, яка дає змогу здійснювати тестування та оцінку мережевих параметрів. Для отримання значень метрик використано розроблений інструмент для генерації трафіку та вимірювання параметрів мережі.

Результати роботи полягають у побудові ефективної мережі, здатної задовольнити вимоги до надійності та продуктивності при використанні систем відеонагляду та IP-телефонії. Важливими аспектами є коректна конфігурація мережевих пристроїв для забезпечення високої пропускної здатності, низької затримки та зниження ймовірності помилок у мережі. Новизна роботи полягає в розробці оптимізованої мережі, що відповідає сучасним вимогам для підприємств, таких як ДП «Антонов», з врахуванням специфіки їх діяльності.

Основними характеристиками мережі є високі показники надійності, безпеки та ефективності використання апаратних ресурсів. Завдяки використанню віртуальних та реальних середовищ для тестування, забезпечено можливість перевірки різних сценаріїв роботи мережі, а також коригування її параметрів відповідно до вимог замовника. Впровадження результатів дослідження дозволяє

забезпечити безперервну роботу систем відеонагляду та IP-телефонії, а також покращити якість обслуговування.

Взаємозв’язок з іншими роботами полягає у використанні сучасних підходів до побудови мереж, інтеграції систем безпеки та ефективної передачі даних. Врахування досвіду попередніх робіт дозволило вибрати оптимальні рішення для побудови надійної мережі, що є основою для подальших розробок у сфері інфраструктури для відеонагляду та IP-телефонії.

Рекомендації щодо використання результатів роботи включають оптимізацію налаштувань мережевих пристроїв, а також можливість розширення та адаптації системи до змінюваних умов діяльності підприємства. Галузь застосування результатів обмежується підприємствами з високими вимогами до надійності, безпеки та якості обслуговування.

Економічна ефективність проекту полягає в оптимізації витрат на апаратні засоби, а також підвищенні загальної ефективності мережевої інфраструктури завдяки вдосконаленню її параметрів.

Значущість роботи полягає у її внеску у вдосконалення процесів побудови надійних та ефективних мереж для сучасних підприємств, а також у можливості адаптації розроблених рішень до інших сфер діяльності.

Прогнозні припущення включають подальший розвиток мережевих технологій, що дозволить досягти ще вищих показників ефективності, а також розширення можливостей для інтеграції нових технологій у цю мережу.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАК, ОДИНИЦЬ І ТЕРМІНІВ

IP – (англ. Internet Protocol) Інтернет протокол;

VoIP – (англ. Voice over Internet Protocol) Голос через Інтернет протокол;

MTU – (англ. Maximum transmission unit) Максимальний розмір блоку корисного навантаження пакету;

ДП – Державне підприємство;

SIP – (англ. Session Initiation Protocol) Протокол встановлення сесії;

КС – Комп’ютерна система;

QoS – (англ. Quality of Service) Якість обслуговування;

SPAN – (англ. Switch Port Analyzer) Локальне дзеркалювання портів;

СМО – Система масового обслуговування;

RTP – (англ. Real-time Transport Protocol) Протокол реального часу;

VBR – (англ. Variable Bitrate) Змінна бітова швидкість;

RAID – (англ. Redundant Array of Independent Disks) Резервний масив незалежних дисків;

ITU – (англ. International Telecommunication Union) Міжнародна спілка електрозв'язку.

ВСТУП

У сучасних умовах стрімкого розвитку інформаційних технологій дедалі більше підприємств стикаються з необхідністю модернізації своїх комп'ютерних систем для підвищення ефективності роботи, забезпечення надійного зв’язку та підтримки високого рівня безпеки. Інформаційні системи нині слугують фундаментом для організації та управління бізнес-процесами, а вдосконалення мережевої інфраструктури стає стратегічним завданням для підприємств у різних галузях. Із зростанням вимог до інформаційної безпеки та надійності систем управління виникає потреба впроваджувати комплексні рішення, які дозволяють не тільки забезпечити ефективний контроль над виробничими процесами, але й оптимізувати внутрішню та зовнішню комунікацію.

Інтеграція систем відеонагляду та IP-телефонії є однією з ключових вимог до інфраструктури сучасного підприємства, оскільки вона дозволяє досягти суттєвого підвищення рівня безпеки та інформаційного контролю. Системи відеоспостереження сприяють підтримці захисту об'єктів, моніторингу виробничих процесів та попередженню інцидентів, тоді як IP-телефонія спрощує комунікаційні процеси, забезпечуючи надійний та зручний зв'язок між підрозділами компанії.

Державне підприємство «Антонов» є одним із провідних авіаційних підприємств України, що має на меті впровадження високоякісних та надійних технологічних рішень для забезпечення безперебійної роботи своїх систем. Підприємство стикається з викликами, пов'язаними з необхідністю постійного вдосконалення мережевої інфраструктури з метою ефективної підтримки операційної діяльності. Одним із важливих завдань для компанії є обґрунтування та вибір оптимальних параметрів мережевих пристроїв, які зможуть забезпечити надійну роботу як систем відеонагляду, так і IP-телефонії.

Сутність завдання полягає в обґрунтуванні параметрів мережевих пристроїв для створення ефективної комп’ютерної системи, що об’єднує відеоспостереження та IP-телефонію. Це включає аналіз вимог до функціональності, продуктивності й

безпеки системи, розробку структурної моделі мережі, вибір відповідного обладнання та його конфігурацію.

Це питання набуває особливої актуальності з огляду на специфіку діяльності ДП «Антонов», що передбачає високі вимоги до безпеки, захисту інформації та безперебійного зв’язку, необхідного для функціонування об’єктів стратегічного значення. Вибір та налаштування відповідних мережевих рішень мають забезпечити високу стабільність, захищеність від можливих загроз і гнучкість інфраструктури, що відповідатиме сучасним вимогам підприємства.

Мета роботи. Метою даної роботи є обґрунтування оптимальних параметрів мережевих пристроїв комп'ютерної системи ДП «Антонов» з урахуванням впровадження системи безпеки відеонагляду та IP-телефонії, а також оцінка їх здатності до обробки зростаючого навантаження без втрати якості обслуговування.

Об’єкт дослідження. Об’єктом дослідження є корпоративна комп'ютерна мережа ДП «Антонов», яка забезпечує функціонування різноманітних систем, включаючи відеонагляд, IP-телефонію, а також інші бізнес-процеси, необхідні для ефективної роботи підприємства.

Предмет дослідження. Предметом дослідження є параметри мережевих пристроїв, що впливають на їхню продуктивність, надійність та якість обслуговування в умовах інтеграції системи безпеки відеонагляду та IP-телефонії.

Методи дослідження. У дослідженні будуть використані такі методи:

Аналіз та синтез – для вивчення існуючої інфраструктури та виявлення проблемних областей.

Моделювання – за допомогою програмних комплексів EVE-NG для створення віртуальної мережі та оцінки навантаження.

Експериментальні дослідження – для вимірювання параметрів продуктивності мережі.

Математичне моделювання – для побудови моделі мережі та оцінки її характеристик у різних сценаріях використання.

1 СТАН ПИТАННЯ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Стан питання

Інформаційні технології є основою функціонування будь-якого сучасного підприємства. Вони забезпечують автоматизацію виробничих процесів, підтримку ефективної комунікації та гарантії інформаційної безпеки. Оскільки виробничі підприємства, такі як ДП «Антонов», працюють у високотехнологічних галузях, вимоги до їхніх комп'ютерних систем стають дедалі вищими. Однією з основних складових комп'ютерної системи є її мережеве обладнання, яке повинно забезпечувати безперебійну передачу даних між усіма підрозділами підприємства.

Для ДП «Антонов» важливим є не тільки забезпечення внутрішньої мережевої інфраструктури, а й інтеграція нових систем, які підвищують рівень безпеки та ефективності комунікації. Серед таких систем – відеонагляд та IP-телефонія, які потребують стабільного функціонування, оскільки від цього залежить оперативність управлінських рішень і безпека підприємства. Проте впровадження цих систем викликає певні труднощі, пов’язані з адаптацією існуючої мережевої інфраструктури до нових вимог. Ці виклики зумовлені не тільки зростанням навантаження на мережеві ресурси, але й потребою в додаткових заходах захисту даних, оптимізації пропускної здатності мережі та мінімізації затримок при передачі інформації.

1.2 Характеристика галузі та особливості застосування комп'ютерних систем у авіації

Авіаційна галузь є одним із ключових рушіїв розвитку сучасного світу, забезпечуючи зв'язок між великими містами та віддаленими регіонами за допомогою авіаліній, аеропортів та систем управління повітряним рухом. Авіація значно впливає на світову економіку: вона підтримує близько 87,7 мільйонів робочих місць та генерує економічний ефект в обсязі 3,5 трильйона доларів щороку, що відповідає рівню 17-ї за величиною економіки світу [1]. Авіаперельоти також сприяють розвитку міжнародного бізнесу та забезпечують зручні умови для

приватних подорожей, адже понад 58% міжнародних туристів обирають повітряний транспорт [2].

Окрім пасажироперевезень, авіація виконує значну роль у вантажоперевезеннях. Її переваги включають швидку доставку, хоча й з обмеженням на обсяг, габарити та вагу вантажу, що збільшує вартість порівняно з морським або автомобільним транспортом. Наприклад, подія з блокадою Суецького каналу у 2021 році показала вразливість альтернативних способів доставки, спричинивши значні економічні втрати через затримку 30% контейнерних перевезень, оцінені в 10 мільярдів доларів щотижня [3].

В Україні літальні апарати використовуються не тільки в цивільній авіації, а й у сільському господарстві (запилення, внесення добрив, моніторинг урожайності) [4] та у оборонних цілях. Наявність власного виробництва літальних апаратів забезпечує країні технологічну перевагу, сприяючи її репутації на міжнародній арені.

Застосування комп'ютерних систем (КС) в авіації сприяє покращенню ефективності та безпеки авіаційних операцій. Основні напрями використання КС в галузі авіації включають:

– Системи управління повітряним рухом: КС забезпечують ефективне управління польотами, включаючи системи наземного контролю, автоматичні системи управління повітряним рухом та зв'язок між літаком і диспетчерами.

– Системи безпеки авіаційних операцій: Для моніторингу та контролю безпеки використовуються КС, що включають системи виявлення перешкод, контролю польоту, ідентифікації літаків тощо.

– Диспетчерські системи: КС використовуються для управління маршрутами, забезпечення диспетчерського зв’язку та моніторингу польотів, сприяючи злагодженій координації авіаційних процесів.

– Авіаційна безпека і кібербезпека: Для захисту авіаційних систем від кіберзагроз використовуються системи виявлення та запобігання вторгнень, шифрування даних, захисту мережевих пристроїв.

– Управління технічним обслуговуванням: Комп'ютерні мережі допомагають організовувати технічне обслуговування літаків, планувати ремонтні роботи, діагностувати несправності та вести облік запасних частин.

Таким чином, КС в авіації мають широке застосування та забезпечують ефективне функціонування всієї галузі. У реальних умовах, їх застосування є гнучким і може змінюватися залежно від потреб і вимог авіаційного підприємства.

1.3 Характеристика і структура об’єкта впровадження

Державне підприємство «Антонов» є перспективним об'єктом для впровадження сучасної комп'ютерної мережі. Це підприємство має усі необхідні ресурси для повного циклу виробництва авіаційної техніки, охоплюючи проєктування, будівництво, випробування, експлуатацію та обслуговування літаків. Завдяки своїм численним досягненням у галузі авіабудування, конструкторське бюро «Антонов» стало відомим створенням унікальних серійних і транспортних літаків, які є знаковими для української та світової авіації.

Відповідно до законодавства України, зокрема Закону «Про розвиток літакобудівної промисловості», авіаційна галузь є однією з пріоритетних у національній економіці України [5]. Основна мета ДП «Антонов» полягає у забезпеченні попиту на пасажирські та вантажні авіаперевезення шляхом створення сучасних, надійних та конкурентоспроможних літальних апаратів, а також у наданні послуг високоякісного технічного обслуговування та ремонту на сучасному рівні.

З огляду на складність технологічних процесів, значні фінансові витрати та високу трудомісткість виробництва авіаційної техніки, підприємства цієї галузі потребують вагомої державної підтримки.

Організаційну структуру ДП «Антонов» представлено на рисунку 1.1.

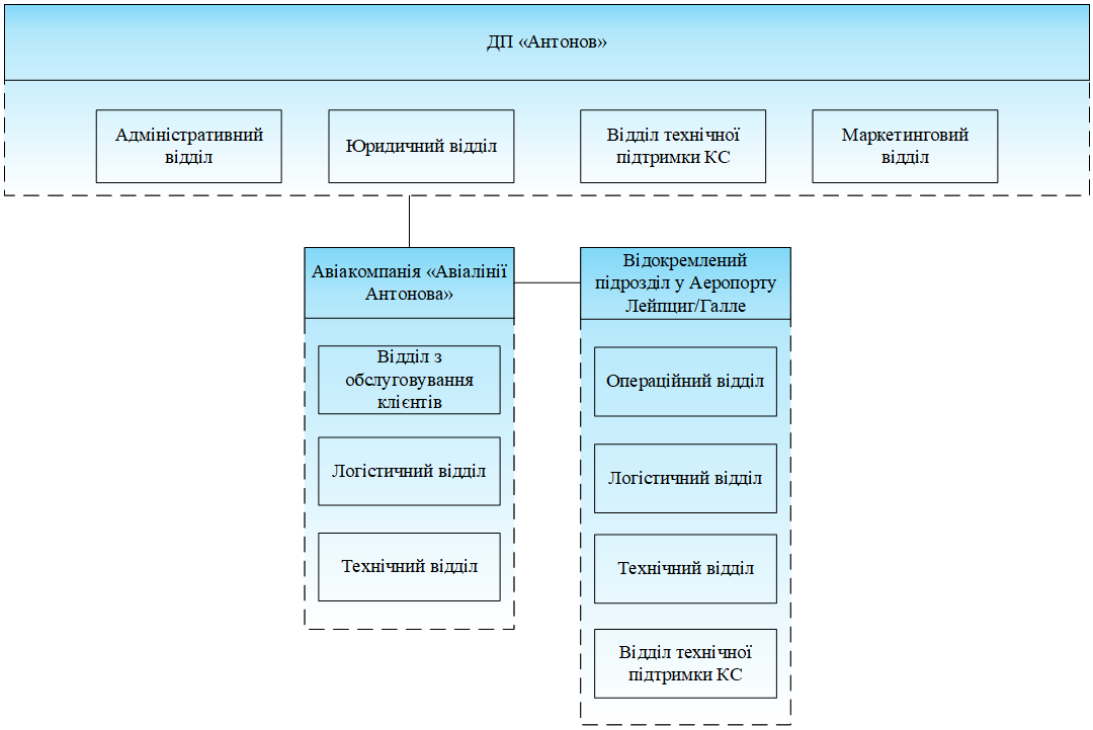


Рисунок 1.1 – Організаційна структура підприємства

Структура організації ДП «Антонов»

Адміністративний відділ головного офісу ДП «Антонов» займається координацією та управлінням різними аспектами внутрішньої діяльності підприємства. Основною його метою є підтримка ефективного функціонування офісу та забезпечення належного виконання адміністративних процесів. Серед його обов'язків – планування й організація офісної роботи, управління постачаннями, організація авіаперевезень, контроль фінансових операцій та ведення бухгалтерського обліку, а також інші адміністративні задачі.

Юридичний відділ відповідає за правову підтримку та супровід діяльності ДП «Антонов». Він надає консультації щодо юридичних питань, підтримує всі підрозділи підприємства з питань укладання угод і контрактів, а також дотримання законодавчих вимог. Крім того, цей відділ займається веденням правової документації, вирішенням суперечок і співпрацею з державними органами та зовнішніми партнерами.

Маркетинговий відділ здійснює розробку і реалізацію маркетингових стратегій компанії. Його основні функції включають дослідження ринку, аналіз

конкурентного середовища, визначення цільової аудиторії та створення маркетингових кампаній. Він також займається рекламою, просуванням продукції та послуг, формуванням позиціонування бренду на ринку, а також ефективним розподілом маркетингових ресурсів.

Відділ технічної підтримки комп'ютерної системи відповідає за підтримку й обслуговування комп'ютерного обладнання підприємства. Він забезпечує встановлення, налаштування і ремонт технічних засобів, підтримку безперебійної роботи корпоративної мережі та захист ІТ-інфраструктури підприємства.

Розташування приміщень та розміщення обладнання головного офісу ДП «Антонов» представлено у додатках А та Б.

Структура організації авіакомпанії «Авіалінії Антонова».

Відділ обслуговування клієнтів авіакомпанії «Авіалінії Антонова» займається підтримкою та задоволенням потреб клієнтів, що використовують послуги вантажоперевезень. Основною метою відділу є забезпечення високоякісного сервісу, оперативне реагування на запити клієнтів та вирішення виникаючих проблем. Відділ обробляє запити, надає інформацію про вантажоперевезення та забезпечує підтримку після здійснення перевезення.

Логістичний відділ авіакомпанії «Авіалінії Антонова» відповідає за організацію та оптимізацію логістичних процесів вантажних перевезень. Він займається плануванням, координацією та моніторингом транспортування вантажів, управлінням складськими операціями, вибором ефективних маршрутів та взаємодією з зовнішніми логістичними партнерами. Головною метою цього відділу є забезпечення безпечної та своєчасної доставки вантажів клієнтам.

Технічний відділ забезпечує консультації з технічних питань, надає підтримку віддаленим ремонтним бригадам та бере участь у плануванні і реалізації проектів, що стосуються технічних аспектів діяльності авіакомпанії. Відділ відповідає за розробку графіків ремонтних робіт, контроль виконання завдань, оновлення документації та проведення аналізу технічних даних. Також відділ здійснює моніторинг параметрів обладнання, перевіряє відповідність документації,

оновлює програмне забезпечення для управління технічними процесами, а також підтримує ефективну роботу вузлів та агрегатів літальних апаратів.

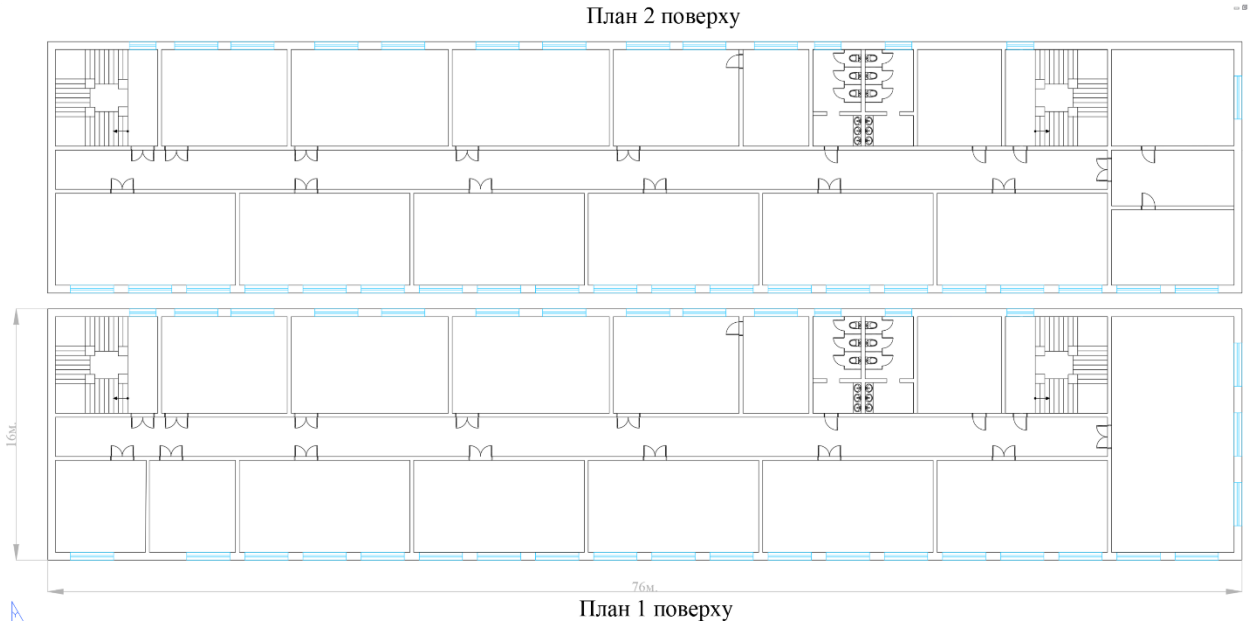


Рисунок 1.2 – План приміщень головного офісу ДП «Антонов»

Структура організації відокремленого підрозділу авіакомпанії «Авіалінії Антонова» в аеропорту Лейпциг/Галле.

Логістичний відділ займається організацією та управлінням вантажними операціями, плануванням і координацією вантажних потоків, а також взаємодією з клієнтами і перевізниками. Відділ відповідає за ефективне використання ресурсів для забезпечення безперебійних та своєчасних перевезень вантажів.

Операційний відділ займається координацією і управлінням усіма операціями, пов'язаними з роботою аеропорту. Це включає взаємодію з аеропортовими службами, організацію посадки, вильоту, технічного обслуговування літаків, контроль вантажу та багажу, а також забезпечення виконання норм безпеки та регуляторних вимог. Відділ сприяє безперебійній роботі аеропорту, співпрацюючи з іншими відділами та службами аеропорту.

Технічний відділ відповідає за планове обслуговування, діагностику та ремонт літаків, їхніх систем і компонентів згідно з технічними стандартами. Відділ також організовує виконання графіків технічних робіт, встановлює пріоритети і

контролює своєчасне виконання завдань. Спільно з іншими відділами забезпечує надійну експлуатацію авіатехніки. Технічний відділ проводить систематичні перевірки, вимірювання і аналіз параметрів обладнання, що дозволяє своєчасно виявляти проблеми і вживати заходів для їх усунення. Крім того, відділ відповідає за контроль якості технічних робіт.

Відділ технічної підтримки КС у відокремленому підрозділі надає технічну підтримку і вирішення проблем, що стосуються комп'ютерних систем у рамках корпоративної мережі, яка функціонує в офісі в Лейпцизі.

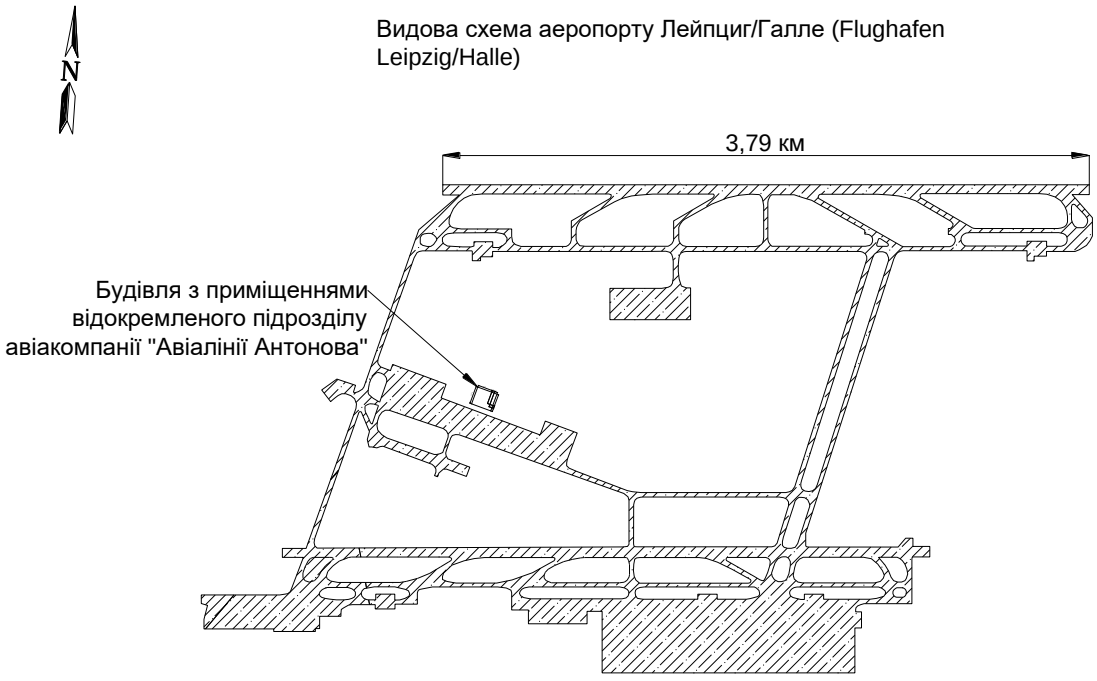


Рисунок 1.3 – Видова схема аеропорту Лейпциг

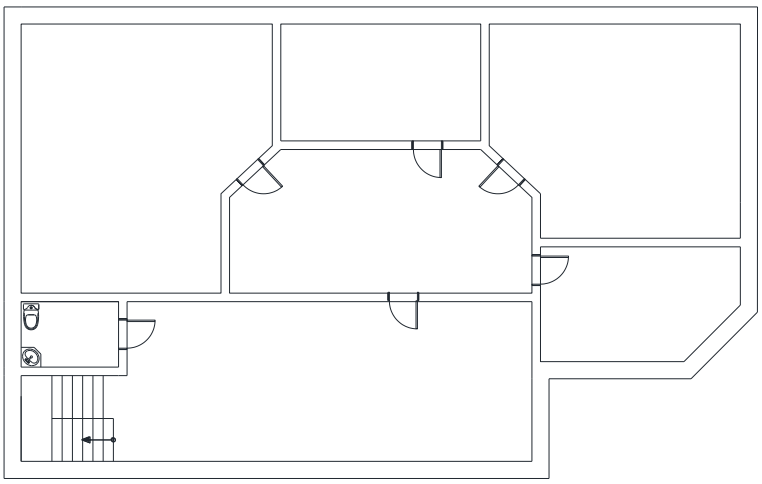


Рисунок 1.4 – План приміщень відокремленого підрозділу ДП «Антонов» у Лейпцигу

1.4 Аналіз існуючих систем телефонії

Існуючі системи телефонії можна поділити на дві основні групи: традиційні аналогові та цифрові рішення на основі IP-телефонії (VoIP). Аналогові телефонні системи – це традиційні системи зв’язку, які функціонують на базі комутованих телефонних мереж (PSTN – Public Switched Telephone Network). У таких системах передача голосу відбувається шляхом електричних сигналів, що безперервно передаються по мідних кабелях. Головним елементом класичних телефонних мереж є комутатори, які забезпечують з’єднання між абонентами, створюючи надійний канал для двостороннього зв'язку [6].

Основні компоненти та архітектура аналогових систем

Аналогові комутатори: Центральним елементом телефонних мереж є комутатор, що відповідає за створення фізичного з’єднання між двома абонентами. Комутатори обробляють кожен дзвінок індивідуально, що забезпечує виділений канал для кожного з’єднання. Комутатори можуть бути як електромеханічними, так і електронними, в залежності від технологічного рівня мережі.

Мідні телефонні лінії: Для передачі голосового сигналу використовується мідний кабель, який забезпечує хорошу провідність сигналу, проте має обмеження

в дальності передачі без підсилення сигналу. Проблеми із шумами та втратою сигналу можуть виникати на довгих відстанях.

Телефонні станції: Для обробки великої кількості з'єднань встановлюються телефонні станції, де з'єднання комутуються між абонентами. Ці станції мають обмежені можливості у масштабуванні, а з’єднання з віддаленими регіонами потребує складної системи маршрутизації.

Сигналізація та комутація: У PSTN системах використовується сигналізація для передачі команд, які ініціюють з'єднання, його підтримку та завершення. Це відбувається через послідовність тонів або імпульсів, що також дозволяє абоненту вводити номери телефону для з’єднання.

Можливості та переваги класичних аналогових телефонних систем

Стабільність та надійність: Завдяки фізичному виділеному каналу аналогові лінії забезпечують стабільне з’єднання без затримок, що гарантує якісний голосовий зв'язок без спотворень або затримок. У разі відсутності інтернету або енергопостачання, аналогові телефонні лінії можуть продовжувати функціонувати завдяки незалежному живленню від телефонної станції.

Простота та зрозуміла архітектура: Використання комутаторів та виділених каналів значно спрощує обслуговування і налаштування таких систем, що робить їх доступними для малих підприємств та офісів. Відсутність потреби у складних мережевих налаштуваннях дозволяє легко використовувати аналогові системи.

Обмеження та недоліки класичних аналогових телефонних систем

Високі витрати на обслуговування: Обслуговування аналогових мереж є дорогим, оскільки вимагає підтримки великої кількості фізичних ліній та обладнання, такого як комутатори. Для масштабування потрібне прокладання додаткових ліній, що значно підвищує витрати.

Мала гнучкість та низька масштабованість: У випадку зростання потреб підприємства розширення аналогових мереж вимагає значних ресурсів. Додавання нових ліній або змінення конфігурацій потребує фізичної реконструкції, що є громіздким і довготривалим процесом.

Обмеженість функціональних можливостей: Класичні телефонні лінії обмежені в можливостях порівняно з IP-телефонією. Вони не підтримують сучасні функції, як-от передача відео, миттєві повідомлення, конференц-дзвінки та інтеграція з іншими цифровими системами. Це ускладнює створення гнучких комунікаційних рішень для сучасних бізнес-потреб.

Якість зв'язку на довгих дистанціях: Для забезпечення зв'язку на великих відстанях аналогові сигнали потребують підсилення, що може викликати спотворення сигналу та шум.

Використання класичних телефонних систем у сучасних умовах

Незважаючи на наявні обмеження, аналогові системи все ще мають застосування в окремих сферах, таких як екстрені служби, медичні заклади, а також у регіонах з обмеженим доступом до інтернету. Завдяки стабільному каналу з'єднання, вони забезпечують надійність зв'язку навіть за несприятливих умов, коли цифрові системи можуть втратити зв'язок.

Таким чином, класичні аналогові системи телефонії залишаються актуальними у вузькоспеціалізованих застосуваннях, але втрачають свою значимість у контексті сучасних потреб бізнесу, який вимагає більшої гнучкості, інтеграції та масштабованості, що може забезпечити лише IP-телефонія.

З розвитком технологій та поширенням IP-мереж популярності набули рішення на основі VoIP, які дозволяють передавати голосові дані через Інтернет або корпоративну мережу, що забезпечує значну економію коштів та відкриває нові можливості для інтеграції. Системи VoIP прості у налаштуванні, масштабуванні і мають додаткові переваги, такі як відеоконференції або служби швидких повідомлень, що сприяє формуванню єдиної комунікаційної платформи. Крім того, VoIP дозволяє використовувати різноманітні пристрої, включаючи комп’ютери, смартфони, спеціалізовані IP-телефони також класичній стаціонарні телефони за наявності спеціальних шлюзів. Широкий спектр доступних для використання пристроїв забезпечує зручність і мобільність зв'язку для співробітників без суттєвого збільшення витрат. Однак, за умови можливості використання відеодзвінків такі системи вимагають високої пропускної здатності, але найбільш

критичним фактором є саме стабільність мережі та висока швидкість обробки повідомлень, оскільки будь-яка затримка або втрата пакетів може суттєво вплинути на якість зв'язку. Для забезпечення якості обслуговування сервісу IP-телефонії спеціалісти мають налаштовувати механізми пріоритезації трафіку, що зменшує затримки передачі та втрати пакетів.

Також важливим аспектом є гарантування безпеки системи телефонії: необхідно застосовувати механізми автентифікації користувачів, пристроїв та додаткових протоколів безпеки, що запобігають несанкціонованому доступу до комунікаційної мережі.

1.5 Аналіз існуючих систем відеонагляду

Системи відеонагляду відіграють важливу роль у гарантуванні безпеки підприємства, оскільки дозволяють здійснювати моніторинг об'єктів та співробітників у режимі реального часу та надають можливість швидко реагувати на небажані або відверто небезпечні ситуації.

Існуючі рішення в сфері відеонагляду включають аналогові системи, які використовують коаксіальні кабелі для передачі відеосигналу, і більш сучасні IP-системи, що передають дані через IP-мережі.

Аналогові системи відеонагляду є відносно недорогими та стабільними, однак їхня функціональність значно обмежена, зокрема через обмежену роздільну здатність зображення та обмежену можливість інтеграції з іншими системами. Для підприємств, що потребують мінімальних витрат, аналогові системи можуть бути прийнятним рішенням, але в умовах сучасних вимог до безпеки вони часто виявляються недостатньо ефективними.

На противагу цьому, IP-відеонагляд пропонує розширені можливості, включаючи високу якість зображення, віддалений доступ до відеокамер через Інтернет, та можливість використання відеоаналітики для автоматичного виявлення руху, розпізнавання облич або об’єктів. IP-системи здатні інтегруватися в корпоративну мережу, що спрощує управління та розширює функціональність, але вони також потребують високих обчислювальних ресурсів і стабільної

пропускної здатності мережі для забезпечення безперервного потоку відеоданих. Відео високої роздільної здатності створює значне навантаження на мережу, тому для ефективного використання таких систем підприємствам необхідно забезпечити надійну інфраструктуру, здатну обробляти великі обсяги трафіку, і налаштувати механізми якості обслуговування для зниження затримок і мінімізації втрат пакетів. Оскільки відеонагляд часто є важливою складовою корпоративної безпеки, захист відеоданих від несанкціонованого доступу також є обов'язковою умовою.

1.6 Технічні виклики при впровадженні систем IP-телефонії

Системи IP-телефонії вимагають від мережі певних умов для забезпечення високої якості зв'язку. Наприклад, для здійснення голосових викликів з використанням кодека g.711 необхідна пропускна здатність каналу становить близько 64 кбіт/с [7]. У той же час для відеодзвінків, особливо у форматі 720p або 1080p, пропускна здатність може зростати до 1-2 Мбіт/с залежно від якості відео [8].

Затримка в передачі даних також є критичним фактором. Ідеальна затримка для голосових викликів повинна бути нижче 150 мс [9], а для відеозв'язку – в межах 300 мс. [10], щоб уникнути помітних проблем зі зв'язком, таких як ехо або затримки.

Щодо безпеки, системи IP-телефонії часто використовують протоколи, такі як SIP (Session Initiation Protocol) з механізмами автентифікації, а також шифрування RTCP (Real-time Transport Control Protocol) для захисту передачі голосових та відео даних.

Відеодзвінки, які стають дедалі популярнішими в бізнес-середовищі, можуть створювати додаткове навантаження на мережу. Системи, які підтримують відеодзвінки, повинні бути спроектовані з урахуванням можливості масштабування та розширення. Наприклад, Zoom та Microsoft Teams пропонують функції адаптивної пропускної здатності, що автоматично регулює якість відео відповідно до наявної пропускної здатності мережі, що забезпечує стабільність з'єднання навіть за умов обмежених ресурсів [11].

Отже, для успішного функціонування систем IP-телефонії важливо забезпечити не лише достатню пропускну здатність, але й оптимізацію трафіку, безпеку, а також можливість підтримки високоякісного відеозв'язку, щоб задовольнити зростаючі вимоги бізнесу.

1.7 Технічні виклики при впровадженні систем відеонагляду

Системи відеонагляду стали важливим елементом сучасної інфраструктури безпеки, особливо для підприємств, що прагнуть забезпечити захист своїх активів. Впровадження таких систем має свої особливості та виклики, які варто враховувати.

Якість та пропускна здатність

Системи IP-відеонагляду пропонують високоякісне відео, яке може досягати роздільної здатності 4K, але це вимагає значних обчислювальних ресурсів та стабільної пропускної здатності. Наприклад, відео у форматі 1080p потребує 1,6-6 Мбіт/с, а при використанні відеоаналітики це навантаження зростає ще більше [12].

Складність інтеграції

Впровадження нових систем часто вимагає інтеграції з існуючими рішеннями. Аналогові системи можуть бути простими у використанні, але їхня функціональність обмежена, тоді як IP-системи забезпечують гнучкість та можливість інтеграції з іншими технологіями, такими як системи управління доступом та відеоаналітика. Однак, інтеграція може вимагати значних інвестицій у нове обладнання та програмне забезпечення.

Безпека даних

Захист відеоданих є критично важливим. Системи IP-відеонагляду піддаються ризикам зловмисних атак, якщо їх не захистити належним чином. Використання шифрування, таких як протокол HTTPS, та автентифікації допомагає знизити ці ризики. У той же час, аналогові системи менш піддаються кіберзагрозам, проте вони не забезпечують такої ж якості та можливостей.

Вартість впровадження

Хоча аналогові системи зазвичай дешевші у початкових витратах, їхня обмежена функціональність може призвести до більших витрат у майбутньому через необхідність частих оновлень або заміни. IP-системи вимагають більше інвестицій на етапі впровадження, але їхня гнучкість та можливість розширення можуть виправдати ці витрати у довгостроковій перспективі.

Технічні вимоги

IP-системи вимагають від підприємства належної мережевої інфраструктури, включаючи швидкісні маршрутизатори та комутатори, здатні обробляти великі обсяги трафіку. Це може бути викликом для організацій, які не мають відповідних ресурсів або досвіду в налаштуванні мереж.

Отже, впровадження систем відеонагляду потребує уважного підходу до оцінки потреб бізнесу, наявної інфраструктури та можливостей забезпечення безпеки. Правильний вибір між аналоговими та IP-системами може значно вплинути на ефективність і надійність системи відеонагляду.

1.8 Аналіз поточної інфраструктури мережевих пристроїв

Аналіз існуючої інфраструктури ДП «Антонов» вказує на те, що мережеві пристрої, які використовуються, переважно відповідають сучасним стандартам, але їхня здатність витримувати значне навантаження потребує додаткового підтвердження. Зокрема, впроваджені системи відеонагляду та IP-телефонії створюють суттєве навантаження на мережу, а дослідження можливих ситуацій, які складно або неможливо було передбачити заздалегідь під час проектування або впровадження систем, стає ключовим для забезпечення стабільності її роботи.

З огляду на вищенаведене можна сформулювати основні проблеми, які потребують вирішення:

– Забезпечення достатньої пропускної здатності мережі. Відеонагляд і IP-телефонія створюють значні обсяги трафіку, які можуть перевищувати можливості поточної мережевої інфраструктури. Для забезпечення безперебійної роботи систем потрібно вдосконалити мережеві пристрої та збільшити пропускну здатність мережі.

– Оптимізація якості обслуговування (QoS). IP-телефонія вимагає високої якості передачі голосових даних, тому потрібно запровадити механізми QoS, щоб забезпечити мінімальні затримки та уникнути втрати пакетів.

– Забезпечення безпеки. Впровадження відеонагляду та IP-телефонії підвищує ризик несанкціонованого доступу до мережі. Необхідно забезпечити захист трафіку за допомогою шифрування та впровадження сучасних засобів автентифікації і фільтрації.

– Масштабованість мережі. Потрібно розробити таку архітектуру мережі, яка зможе легко розширюватися у разі збільшення кількості пристроїв або нових потреб підприємства.

Ці проблеми вимагають детального вивчення та обґрунтованого підходу до вибору параметрів мережевих пристроїв, щоб забезпечити надійну роботу як поточних систем, так і тих, що будуть впроваджені в майбутньому.

1.9 Постановка завдання дослідження

Для проведення дослідження необхідно:

1. Провести аналіз навантаження на мережу

У сучасному середовищі, де впроваджуються системи відеоспостереження та IP-телефонії, важливо провести детальний аналіз навантаження на мережу. Це дозволить забезпечити належний рівень продуктивності та стабільності. Це включає:

– Моделювання мережевої топології: Розглянути можливість побудови моделі мережі, з можливістю підключення апаратного мережевого обладнання у віртуальну інфраструктуру підприємства. Метою є створення моделі віртуальної мережі, яка дозволить оцінити їх взаємодію в умовах, наближених до реальних, а також дослідити ефективність та продуктивність мережі при різних параметрах конфігурації.

– Сценарії навантаження: Необхідно змоделювати різноманітні сценарії використання, які включають одночасні підключення, трансляцію відео з декількох

камер та інші типи трафіку. Це дозволить отримати точні дані про обсяги трафіку, який генерують ці системи в умовах реальної експлуатації.

– Аналіз результатів: На основі отриманих даних про навантаження на мережу (пропускна здатність, затримка, втрата пакетів) провести оцінку максимальної кількості одночасних з'єднань, які може витримати мережа без значного погіршення якості обслуговування.

2. Визначити часу обробки повідомлення на вузлах

Одним з важливих аспектів проектування мережі є оцінка часу, необхідного для обробки повідомлень. Для цього необхідно:

– Провести вибірку вузлів для аналізу: Спочатку визначаються основні вузли мережі, через які проходить найбільший обсяг трафіку або які відповідають за критично важливі функції. Це можуть бути комутатори, маршрутизатори, сервери IP-телефонії тощо.

– Налаштувати системи моніторингу: Для збору даних про затримки можуть використовуватися функції моніторингу, як SPAN (Switched Port Analyzer) на комутаторах Cisco, яка дозволяє дублювати трафік з вибраного порту на аналізуючий пристрій за допомогою програми Wireshark. Це допомагає відстежити час проходження пакетів через вузли.

– Збір і аналіз трафіку: На цьому етапі проводиться фіксація часу обробки різних типів пакетів (наприклад, сигналізації SIP або потоків RTP для голосового та відеотрафіку). Під час обробки вимірюються показники, такі як затримка передачі пакетів, джиттер, і частота втрат.

– Аналіз результатів: На основі зібраних даних визначається середній час обробки на кожному вузлі та загальна затримка у мережі. Аналіз дозволяє виявити вузли з найбільшими затримками та прийняти рішення щодо оптимізації налаштувань мережевих пристроїв або навіть їх модернізації.

3. Здійснити проєктування математичної моделі мережі

Розробка математичної моделі мережі дозволить прогнозувати її поведінку в умовах різних навантажень.

Опис моделі: Визначити мережу як замкнуту систему масового обслуговування (СМО), що включає основні параметри, такі як інтенсивність потоку трафіку, кількість обслуговуючих вузлів та середні часи обслуговування.

Моделювання сценаріїв: Використовувати різні сценарії навантаження для моделювання роботи мережі у різних умовах. Це дозволить оцінити ймовірність черг, середній час очікування, завантаження вузлів та інші критично важливі показники.

Оптимізація параметрів: На основі результатів моделювання визначити оптимальні параметри мережі, які забезпечать її ефективність і надійність.

4. Розробка рекомендацій

На основі проведених досліджень та аналізу, необхідно підготувати рекомендації щодо:

Оптимальних параметрів мережевих пристроїв: Визначити, які конкретні мережеві пристрої та їх налаштування забезпечать найкращу продуктивність для систем відеонагляду та IP-телефонії.

2 ТЕОРЕТИЧНА ЧАСТИНА

2.1 Теоретичне обґрунтування мережевих параметрів для інтеграції відеоспостереження та IP-телефонії

2.1.1 Метод вирішення

Для досягнення поставлених завдань застосовуються такі підходи як:

– Моделювання мережі за допомогою програмних комплексів;
– Збір та аналіз мережевих параметрів;
– Комп'ютерне і статистичне імітаційне моделювання.

2.1.1.1 Моделювання мережі: програмні комплекси моделювання

У процесі дослідження мережевих параметрів одним із ключових методів є моделювання. Це дозволяє створити віртуальне середовище, в якому можна тестувати різні конфігурації, перевіряти продуктивність і аналізувати поведінку системи за різних умов. Для моделювання використовуються спеціалізовані програмні комплекси, кожен із яких має свої особливості.

Cisco Packet Tracer є простим у використанні симулятором, що підходить для створення базових мережевих топологій та їх тестування. Його перевага полягає в наочності та можливості працювати з обмеженим набором пристроїв, які повністю віртуалізовані. Це робить програму чудовим інструментом для початкового етапу проектування мережі. Однак, Packet Tracer не підтримує роботу з реальними мережевими образами, що обмежує його використання для дослідження складних сценаріїв.

EVE-NG, на відміну від Packet Tracer, є потужною платформою для віртуалізації реальних мережевих образів. Цей комплекс дозволяє створювати мережі з використанням програмного забезпечення пристроїв, таких як Cisco, Juniper, або інших виробників. Завдяки цьому результати моделювання є більш наближеними до реальних умов. Крім того, EVE-NG забезпечує інтеграцію із зовнішніми інструментами для аналізу трафіку, такими як Wireshark. Недоліком

може бути більша складність у використанні та потреба в потужних апаратних ресурсах. Особливістю EVE-NG можливість створення віртуальних мереж, які можуть бути інтегровані з реальними мережами, включно з Інтернетом. Це досягається через використання різних методів з'єднання, таких як підключення до фізичних мережевих інтерфейсів або налаштування віртуальних маршрутизаторів та комутаторів для забезпечення взаємодії з зовнішніми мережами. Завдяки цьому, можна моделювати реальні умови з підключенням до Інтернету, що дає змогу тестувати роботу мережі в умовах реального трафіку та оцінювати її поведінку при інтеграції з існуючими інфраструктурами. Це робить EVE-NG потужним інструментом для створення та тестування гібридних мереж, які поєднують віртуальні та фізичні елементи.

Mininet спеціалізується на моделюванні програмно-конфігурованих мереж (SDN). Цей інструмент дозволяє швидко розгортати мережеві сценарії з великим числом вузлів та тестувати алгоритми маршрутизації чи керування трафіком. Проте Mininet має обмеження у відтворенні класичних мережевих пристроїв, що робить його менш універсальним для завдань, пов’язаних із IP-телефонією або відеонаглядом.

Отже, для завдань проектування та тестування системи безпеки відеонагляду та IP-телефонії оптимальним вибором є поєднання Cisco Packet Tracer для швидкої візуалізації і тестування основної топології та EVE-NG для моделювання реалістичних сценаріїв із використанням мережевих образів.

2.1.1.2 Мережеві пристрої та їх застосування для тестування мережі

Окрім програмного моделювання, для вивчення параметрів мережі важливу роль відіграють апаратні засоби та методи їх використання. Реальні пристрої забезпечують точні дані про поведінку мережі, дозволяють тестувати специфічні конфігурації та отримувати результати в умовах, наближених до реальних.

Одним із ключових механізмів є використання комутаторів, маршрутизаторів і шлюзів, які забезпечують передачу даних між вузлами. Ці пристрої дозволяють організувати тестове середовище, де можна вимірювати

затримки, джиттер, втрати пакетів, а також продуктивність у різних умовах навантаження.

Для аналізу мережевого трафіку широко застосовується механізм дзеркалювання портів (SPAN або RSPAN) [13]. Він дозволяє перехоплювати пакети, які проходять через задані порти мережевих пристроїв, і направляти їх до системи збору даних. На основі цих даних за допомогою програмного забезпечення, такого як Wireshark, проводиться аналіз параметрів протоколів, зокрема RTP, що використовується для передачі голосу та відео.

Таким чином, використання апаратного забезпечення дозволяє проводити комплексні дослідження мережевих параметрів, перевіряти роботу мережі в умовах реального навантаження та оптимізувати її функціонування для критичних застосувань, таких як IP-телефонія та системи відеоспостереження.

2.1.1.3 Комп'ютерне і статистичне імітаційне моделювання

Комп'ютерне і статистичне імітаційне моделювання є важливим інструментом для детального вивчення та аналізу складних систем, зокрема комп'ютерних мереж, інформаційних технологій, бізнес-процесів та інших галузей, де велика кількість змінних і взаємодій потребує комплексного підходу. Застосування цього методу дозволяє створювати віртуальні моделі реальних систем, що дає змогу детально аналізувати їхню роботу, прогнозувати результати функціонування та знаходити способи підвищення ефективності.

Імітаційне моделювання надає можливість відтворювати і досліджувати різноманітні сценарії, що імітують поведінку системи під дією заданих параметрів і умов. Це дає змогу заздалегідь оцінювати, як система реагуватиме на зміни в умовах експлуатації, виявляти оптимальні шляхи її розвитку, а також виявляти потенційні ризики або недоліки в її конструкції чи управлінні. Такий підхід сприяє глибшому розумінню того, як функціонує система, дозволяє вдосконалювати її, роблячи більш адаптованою до непередбачуваних ситуацій.

Статистичне імітаційне моделювання в свою чергу застосовує статистичні методи для детального аналізу результатів моделювання. Зокрема, теорія

ймовірностей, регресійний аналіз, аналіз варіацій і багато інших статистичних методів використовуються для обробки та інтерпретації зібраних даних, що дозволяє отримувати не лише точні, а й обґрунтовані висновки. Цей процес допомагає виявляти закономірності, оцінювати ймовірність настання подій, аналізувати тренди і проводити порівняння з іншими сценаріями.

Завдяки поєднанню комп'ютерного та статистичного підходів, імітаційне моделювання дозволяє здійснювати багатосторонній аналіз, підвищувати ефективність системи, приймати обґрунтовані рішення і значно зменшувати ризики. У середовищах з високою невизначеністю та динамікою, таких як сучасні комп'ютерні мережі, такий підхід є надзвичайно корисним для забезпечення надійності та стійкості роботи системи, оскільки надає можливість здійснювати адаптивне управління в реальному часі.

Задачі, що пов'язані з аналізом замкнутих мереж масового обслуговування, часто вирішуються за допомогою комп'ютерного моделювання. У таких мережах кількість одиниць у системі, тобто кількість пакетів, фіксована, і вони циркулюють між різними вузлами. Завдяки симуляціям можна оцінити ефективність мережі, її параметри та зрозуміти, як поведінка мережі змінюється під різними умовами. Однак ці симуляційні моделі можуть бути складними у реалізації та аналізі, тому є необхідність у використанні математичних моделей, які дозволяють спростити цей процес та зробити результати більш загальними та застосовними до широкого кола задач.

Одним із основних математичних інструментів для аналізу замкнутих мереж є алгоритм Бузена, який дозволяє розраховувати характеристику мережі, зокрема очікувану кількість пакетів у кожному з вузлів, час їх перебування в мережі та інші параметри. Алгоритм дає змогу моделювати ситуацію, коли кількість одиниць у системі постійна, і вони циркулюють між різними вузлами з певними ймовірностями. Цей метод дозволяє отримати точні результати, не вдаючись до складних симуляцій, що робить його ідеальним для широкого застосування [14].

У даному прикладі розглядається замкнена мережа масового обслуговування, що складається з кількох вузлів, через які передаються пакети

даних. Ця модель дозволяє оцінити характеристики мережі, такі як навантаження, часи затримок, ймовірність втрат пакетів та ефективність обробки трафіку на різних етапах.

Мережа складається з M вузлів, кожен з яких має певну пропускну здатність для обробки пакетів. Кожен вузол обробляє пакети з певною ймовірністю, і після обробки пакети можуть переміщатися на інші вузли за визначеними ймовірностями. Ці ймовірності переходу між вузлами визначають ефективність маршрутизації та навантаження на мережу.

Термінологія:

n_i – кількість одиниць, присутніх на і-му об’єкті

M – кількість об’єктів або центрів $\sum_{i=1}^{M} n_i = N$

$1/\mu_i$ – час обслуговування для і-го об'єкта

P_{ij} – ймовірність того, що одиниця здійснить перехід від і-го до j-го елементу протягом i, j = 1,2,..., M

G(N) – нормалізуюча константа

X_i – дійсний позитивний розв’язок рівнянь типу власного вектора

θ_i – відносне використання і-го вузла

$E[n_i]$ – очікувана кількість одиниць у і-му вузлі

W_i – очікуваний час очікування у і-му вузлі

α_1, α_2, α_3 – перехідні ймовірності.

Модель замкненої мережі масового обслуговування дозволяє аналізувати потоки трафіку та обчислювати різні параметри мережі, включаючи затримки, навантаження на вузли, ймовірність втрат пакетів та інші важливі характеристики. За допомогою такої моделі можна оптимізувати роботу мережі та забезпечити ефективну передачу даних навіть за високих навантажень.

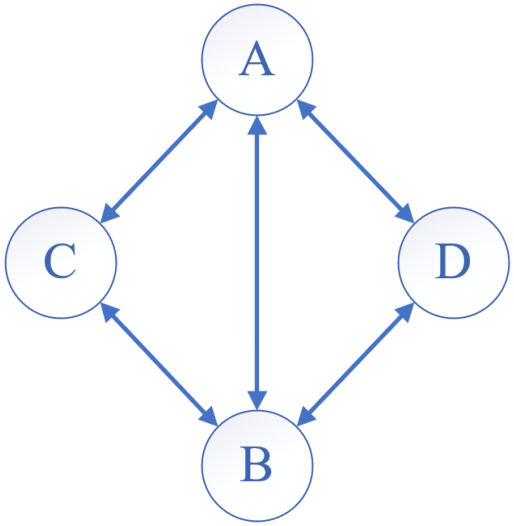


Рисунок 2.1 – Графова модель комп'ютерної мережі

Мережа складається з чотирьох вузлів (A, B, C, D), між якими існують спрямовані зв'язки. Кожен вузол відповідає за обробку пакетів, і зв'язки між ними характеризуються ймовірністю передачі пакетів.

Визначення ймовірностей передачі пакетів.

Задаємо ймовірність передачі пакета між кожним з вузлів. Вона подається у вигляді матриці ймовірностей переходів P, де кожен елемент P_{ij} – це ймовірність того, що пакет, що надійшов у вузол i, буде переданий до вузла j.

$$P = \begin{pmatrix} P_{AA} & P_{AB} & P_{AC} & P_{AD} \\ P_{BA} & P_{BB} & P_{BC} & P_{BD} \\ P_{CA} & P_{CB} & P_{CC} & P_{CD} \\ P_{DA} & P_{DB} & P_{DC} & P_{DD} \end{pmatrix},$$

При цьому, ймовірність того, що вузол зв'язується сам з собою P_{ii} дорівнює 0.

Матриця обробки пакетів.

Кожен вузол має певний час на обробку пакета. Час обробки μi для вузла i визначає, скільки часу потрібно для обробки одного пакета.

Матриця обробки пакетів може бути представлена як:

$$T = \begin{pmatrix} T_A \\ T_B \\ T_C \\ T_D \end{pmatrix},$$

де T_A, T_B, T_C, T_D – це час обробки пакета в вузлах A, B, C та D відповідно.

Модель масового обслуговування для кожного вузла.

Кожен вузол мережі можна трактувати як систему масового обслуговування (queueing system), де пакети, що надходять у вузол, обробляються по черзі. Для таких систем розраховуються стандартні параметри, зокрема:

– Інтенсивність потоку пакетів (λ_i): кількість пакетів, що надходять у вузол і за одиницю часу.

– Час перебування пакета в системі (W_i): середній час, що пакет проводить у вузлі і (включаючи час обробки та час очікування в черзі).

– Кількість пакетів в черзі (N_i): середня кількість пакетів, що знаходяться в обробці або в черзі на обробку у і-му вузлі.

Ці параметри для кожного вузла можна розрахувати, використовуючи теорію масового обслуговування.

Метод Бузена в контексті комп'ютерного та статистичного імітаційного моделювання зазвичай використовується для моделювання складних систем і процесів, де враховуються випадкові зміни і стохастичні явища. Це метод, який застосовується для оцінки ймовірностей або прогнозування результатів в умовах невизначеності.

Зазначений метод має широкий спектр використання в різних сферах, зокрема для прогнозування поведінки мереж, оцінки ризиків, розробки стратегії управління та оптимізації технічних систем.

2.2 Протоколи і технології в передачі мультимедійних даних

Основу передачі мультимедійних даних через мережі складають протоколи сигналізації та передачі даних, які забезпечують встановлення, підтримку та завершення сеансів зв’язку, а також надійну і ефективну передачу голосу, відео та

інших мультимедійних даних. Ці протоколи широко використовуються не тільки в IP-телефонії та системах відеонагляду, але й у мультимедійних сервісах та інших мережевих застосунках.

Основними протоколами для забезпечення мультимедійних сеансів є SIP (Session Initiation Protocol), H.323 та RTP (Real-time Transport Protocol). Протокол SIP є стандартом для ініціалізації, зміни та завершення мультимедійних сеансів, включаючи голосові і відео-виклики. H.323, хоч і є більш раннім стандартом та зберігає популярність в корпоративних мережах для передачі мультимедійних даних, проте з появою та широким поширенням SIP поступово втрачає популярність у сучасних мережах. RTP забезпечує передачу мультимедійних даних в режимі реального часу з підтримкою синхронізації між потоками даних.

2.2.1 Протокол встановлення сесії

Session Initiation Protocol (SIP) – це стандартний протокол сигналізації, розроблений IETF (Internet Engineering Task Force) у RFC 3261 для ініціації, управління та завершення сеансів зв’язку в IP-мережах [15]. SIP часто використовується для голосових та відеодзвінків, а також для багатокористувацьких конференцій, миттєвих повідомлень, передавання даних, ігор тощо. Протокол підтримує текстовий формат, що полегшує його розуміння, налагодження та інтеграцію з іншими інтернет-протоколами.

Основними функціями SIP є ініціація, модифікація та завершення сеансів [15]:

– Ініціація сеансу: SIP встановлює з’єднання між двома або більше кінцевими точками для обміну медіа-даними.

– Модифікація сеансу: Протокол дозволяє змінювати параметри зв’язку під час сеансу, наприклад, додавати нових учасників або перемикатися з аудіо на відео.

– Завершення сеансу: SIP забезпечує належне завершення зв’язку, звільняючи ресурси мережі.

Особливості та переваги SIP:

– Гнучка адресація: SIP використовує SIP URI для адресації (наприклад, sip:username@domain.com), що забезпечує сумісність з іншими інтернет-протоколами.

– Незалежність від транспортного рівня: SIP працює поверх TCP, UDP або інших транспортних протоколів.

– Підтримка NAT та фаєрволів: SIP має механізми для обробки NAT, що полегшує підключення через інтернет.

– Розширення та адаптивність: SIP підтримує різні медіа-протоколи (зазвичай RTP) та дозволяє інтегрувати нові функції завдяки гнучкій архітектурі розширень.

Недоліки SIP:

– Проблеми із сумісністю: Різні реалізації SIP можуть не повністю сумісні між собою через варіативність інтерпретацій протоколу.

– Складність при використанні NAT: Незважаючи на підтримку NAT, SIP потребує додаткових налаштувань для проходження через мережеві адреси, що транслюються.

– Безпека: SIP-повідомлення можуть бути вразливі до атак, таких як перехоплення та підробка, тому рекомендується використовувати захист через TLS і SRTP для захищених з’єднань.

Для забезпечення захисту SIP використовує TLS (Transport Layer Security) для шифрування сигналізації, а також SRTP (Secure Real-time Transport Protocol) для шифрування медіа-потоків [16]. Крім того, SIP може використовувати автентифікацію за допомогою HTTP Digest Authentication, щоб підтвердити особу абонента [17].

Архітектура SIP базується на чотирьох основних компонентах [15]:

User Agent (UA) – пристрій або програма, що виступає як кінцева точка для ініціювання, приймання або обробки викликів. Він має дві підкомпоненти:

User Agent Client (UAC) – створює та надсилає запити іншим UA.

User Agent Server (UAS) – приймає запити та надсилає відповіді.

Проксі, проксі-сервер (Proxy Server) – посередницька одиниця, що виконує роль як сервера, так і клієнта з метою формування запитів від імені інших клієнтів. Основна роль проксі-сервера полягає в маршрутизації, що означає, що його завдання – гарантувати, що запит надсилається іншій одиниці, найближчій до цільового користувача. Проксі також корисні для дотримання політики (наприклад, для забезпечення того, чи дозволено користувачу здійснювати дзвінок). Проксі інтерпретує, а при необхідності перезаписує певні частини повідомлення запиту перед його пересиланням.

Сервер реєстрації (Registrar Server) – зберігає інформацію про поточне місцезнаходження UA, що дозволяє користувачам бути доступними на різних пристроях і в різних місцях.

Сервер перенаправлення (Redirect Server) – це сервер, який виступає в ролі агента користувача і генерує відповіді з кодом 3xx на отримані запити, вказуючи клієнту звертатися до альтернативного набору URI.

SIP-повідомлення

SIP працює за принципом обміну запитами та відповідями, аналогічно до HTTP. Запити ініціюються клієнтською стороною (UAC), а сервер відповідає, обробляючи запит. У таблиці 2.1 наведені основні повідомлення SIP.

Таблиця 2.1 – Основні типи повідомлень SIP

Запити	
INVITE	Ініціює новий сеанс зв’язку, наприклад, виклик
ACK	Підтверджує завершення встановлення сеансу після отримання відповіді на INVITE
BYE	Завершує активний сеанс
CANCEL	Відміняє встановлення з’єднання до його завершення
OPTIONS	Запитує інформацію про можливості іншого UA (наприклад, підтримувані кодеки)

Кінець таблиці 2.1

Запити	
REGISTER	Реєструє пристрій користувача на сервері реєстрації, що дозволяє іншим користувачам знайти його
Відповіді	
1xx (Informational)	Попередні відповіді, які вказують, що запит опрацьовується
2xx (Success)	Успішні відповіді, що підтверджують успішну обробку запиту
3xx (Redirection)	Відповіді, що перенаправляють клієнта до іншого пункту
4xx (Client Error)	Помилки клієнта (наприклад, неправильний запит або відсутність доступу)
5xx (Server Error)	Помилки сервера (невдалося обробити запит через внутрішні проблеми)
6xx (Global Failure)	Фатальні помилки (запит не може бути виконаний у будь-якому випадку)

На рисунку 2.2 показаний стандартний процес сигналізації у протоколі SIP (Session Initiation Protocol), що включає ініціацію, встановлення, передачу даних (RTP) та завершення дзвінка між двома SIP-клієнтами через SIP-сервер.

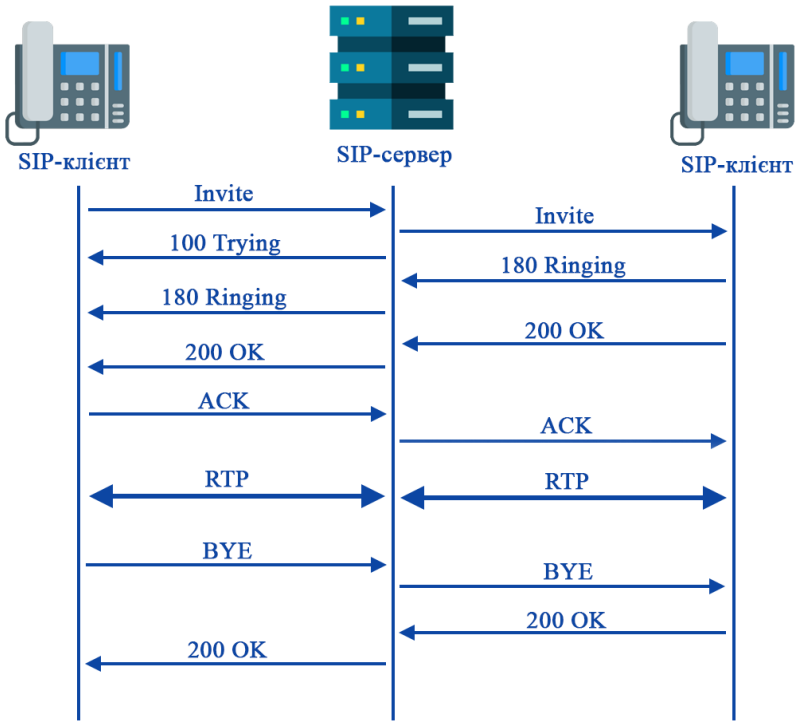


Рисунок 2.2 – Приклад обміну повідомленнями SIP для встановлення та завершення прямого медіа-сеансу.

Ось як відбувається кожен етап:

1. Ініціація виклику.

SIP-клієнт, що ініціює виклик, надсилає повідомлення INVITE до SIP-сервера. Сервер обробляє цей запит та пересилає його на інший SIP-клієнт (отримувача). Після отримання запиту сервер відповідає ініціатору кодом стану 100 Trying, що підтверджує прийом запиту.

2. Очікування відповіді.

Коли дзвінок доходить до отримувача, отримувач надсилає SIP-серверу повідомлення 180 Ringing, а він в свою чергу надсилає його ініціатору, що сигналізує про те, що телефон отримувача дзвонить.

3. Прийняття виклику.

Коли отримувач відповідає на дзвінок, надсилається відповідь 200 OK, яка проходить через SIP-сервер і доходить до ініціатора. Це означає, що дзвінок готовий до встановлення.

4. Підтвердження виклику.

Ініціатор відповідає на 200 OK повідомленням ACK, підтверджуючи, що виклик встановлено успішно.

5. Передача голосу/відео (сесія RTP).

Після підтвердження обидва SIP-клієнти починають обмін голосовими даними через RTP (Real-Time Transport Protocol). Цей сеанс може проходити як через сервер, якщо, наприклад, необхідно записувати дзвінки, так і минаючи його.

6. Завершення виклику.

Коли один із клієнтів хоче завершити виклик, він надсилає повідомлення BYE. Інший клієнт підтверджує завершення, відповідаючи повідомленням 200 OK.

Синтаксис SIP-повідомлень базується на текстових заголовках і нагадує повідомлення HTTP/1.1 проте, не SIP не є розширенням HTTP [15]. Наприклад, INVITE-запит має структуру подану на рисунку 2.3.

```
Session Initiation Protocol (INVITE)
    Request-Line: INVITE sip:1004@192.168.44.78;user=phone SIP/2.0
        Method: INVITE
        Request-URI: sip:1004@192.168.44.78;user=phone
            Request-URI User Part: 1004
            Request-URI Host Part: 192.168.44.78
        [Resent Packet: False]
    Message Header
        Via: SIP/2.0/UDP 192.168.44.182:5060;branch=z9hG4bKb2a673744d4e8
            Transport: UDP
            Sent-by Address: 192.168.44.182
            Sent-by port: 5060
            Branch: z9hG4bKb2a673744d4e8
        From: 1003 <sip:1003@192.168.44.78;user=phone>;tag=3655371753
            SIP from display info: 1003
            SIP from address: sip:1003@192.168.44.78;user=phone
                SIP from address User Part: 1003
                SIP from address Host Part: 192.168.44.78
                SIP From URI parameter: user=phone
            SIP from tag: 3655371753
        To: <sip:1004@192.168.44.78;user=phone>
            SIP to address: sip:1004@192.168.44.78;user=phone
                SIP to address User Part: 1004
                SIP to address Host Part: 192.168.44.78
                SIP To URI parameter: user=phone
        Call-ID: 3504225267@192.168.44.182
        [Generated Call-ID: 3504225267@192.168.44.182]
        CSeq: 1 INVITE
            Sequence Number: 1
            Method: INVITE
        Contact: 1003 <sip:1003@192.168.44.182:5060;user=phone;transport=udp>
            SIP C-URI display info: 1003
            Contact URI: sip:1003@192.168.44.182:5060;user=phone;transport=udp
                Contact URI User Part: 1003
                Contact URI Host Part: 192.168.44.182
                Contact URI Host Port: 5060
                Contact URI parameter: user=phone
                Contact URI parameter: transport=udp
        User-Agent: Cisco ATA 186  v3.2.1 atasip (050616A)
```

Рисунок 2.3 – Структура повідомлення INVITE

У вищенаведеному прикладі SIP-повідомлення INVITE ініціює виклик до користувача 1004 за IP-адресою 192.168.44.78 (тип: телефон). Заголовок Via вказує

маршрут через 192.168.44.182:5060 (UDP), а From ідентифікує відправника як 1003 з IP 192.168.44.78. Заголовок To містить інформацію про отримувача 1004 за тією ж IP-адресою. Унікальний ідентифікатор виклику задано через Call-ID (3504225267@192.168.44.182), а порядок запиту вказаний у CSeq (номер 1). Контактна адреса відправника для відповіді: sip:1003@192.168.44.182:5060, з параметрами user=phone і transport=udp.

2.2.2 Транспортний протокол реального часу

Протокол RTP (англ. Real-time Transport Protocol) – це мережевий протокол, який використовується для передачі даних у реальному часі, таких як аудіо, відео або інші мультимедійні дані. Він був розроблений для роботи з додатками, що вимагають низької затримки та підтримують високий рівень синхронізації. RTP визначений у RFC 3550 і часто використовується разом із протоколом управління RTP (RTCP), що дозволяє здійснювати моніторинг якості передачі даних [18].

Основні характеристики RTP:

– Передача даних у реальному часі. RTP забезпечує доставку мультимедійних даних з мінімальними затримками.

– Ідентифікація даних. RTP надає можливість маркувати пакети для забезпечення коректної реконструкції мультимедійного потоку.

– Синхронізація. Використання часових міток та ідентифікаторів синхронізації (SSRC) дозволяє узгоджувати окремі мультимедійні потоки.

Структура пакета RTP

Пакет RTP складається з заголовка і даних (корисного навантаження). Структура заголовка має формат наведений у таблиці 2.2.

Таблиця 2.2 – Структура заголовка протоколу RTP

Поле	Розмір (біт)	Опис
Version (V)	2	Версія протоколу (поточна версія – 2).
Padding (P)	1	Індикатор наявності додаткових байтів в кінці пакета.
Extension (X)	1	Вказує на наявність додаткового заголовка.

Кінець таблиці 2.2

Поле	Розмір (біт)	Опис
CSRC Count (CC)	4	Кількість ідентифікаторів CSRC.
Marker (M)	1	Спеціальний маркер, що визначає важливість пакета.
Payload Type (PT)	7	Тип корисного навантаження (кодек).
Sequence Number	16	Порядковий номер пакета для виявлення втрат та відновлення порядку.
Timestamp	32	Часова мітка для синхронізації.
SSRC	32	Унікальний ідентифікатор джерела потоку.
CSRC List	0-15x1032	Список ідентифікаторів джерел (опціонально).

Протокол RTP не надає всі можливі функції. Найбільш помітні обмеження RTP:

– Відсутність гарантій якості обслуговування (QoS). Хоча RTP розроблений для сценаріїв, чутливих до затримок, він не має вбудованих функцій, які забезпечують QoS. RTP лише передає інформацію, яка дозволяє реалізувати QoS на інших рівнях стеку.

– Не займається виділенням чи резервуванням ресурсів. RTP не відповідає за управління ресурсами, які можуть бути необхідні для роботи, наприклад, смуга пропускання чи пріоритет трафіку. Це завдання покладається на інші протоколи чи механізми мережі.

Ці обмеження компенсуються інтеграцією RTP з іншими технологіями для створення комплексних рішень передачі даних у реальному часі.

2.3 Огляд метрик якості обслуговування в комп’ютерних мережах

Якість функціонування мережі визначається кількома основними параметрами, кожен з яких впливає на ефективність передачі даних та задоволення потреб користувачів. Розуміння цих параметрів дозволяє не лише оцінити поточний стан мережі, але й визначити заходи для покращення її продуктивності.

Швидкість передачі даних (Throughput): Цей параметр характеризує фактичну швидкість передачі даних, з якою стикається користувач під час роботи з мережею.

Пропускна здатність (Bandwidth): Цей параметр характеризує обсяг даних, які можуть передаватися через мережевий канал за одиницю часу, тобто «ширину» або ємність каналу передачі даних. Вимірюється в бітах на секунду (bps) і є ключовим фактором, який визначає, наскільки швидко мережа може обробляти великі обсяги трафіку.

Різниця між пропускною здатністю і швидкістю.

Пропускна здатність визначає ємнісний параметр каналу обґрунтовуючись теоретично, тоді як швидкість задає кількісну характеристику, яку можна визначити емпірично, тобто вона залежить від поточних умов роботи з’єднання. Наприклад, якщо канал має широку смугу пропускання, але при цьому в конкретний момент часу присутнє високе мережеве навантаження спричинене конкуренцією за канали передачі, недостатньою обчислювальною спроможністю серверів або мережевого обладнання, фактична швидкість з'єднання може бути значно нижчою від заявленого номіналу [19].

Затримка (Delay, Latency): Затримка це параметр що описує час, потрібний для того, щоб пакет даних пройшов від відправника до отримувача. Чим менша затримка, тим менше часу на відгук потребується при роботі в мережі. Термін затримка часто використовується як синонім із латентністю, але між ними є тонка різниця.

Латентність, від лат. latentis – прихований, невидимий, це затримка мережі що стосується загального часу, необхідного для надсилання всього повідомлення, тоді як затримка розповсюдження (Propagation delay) означає час, необхідний для проходження першого біта по каналу між відправником і одержувачем [20].

Затримка розповсюдження описується формулою поданою нижче,

$$d/s,$$

де d – відстань, а s – це швидкість поширення хвилі. Затримка вимірюється в мілісекундах (ms), і її зниження є пріоритетом для додатків реального часу, таких як IP-телефонія або відеозв'язок. На рисунку 2.4 подано діаграму внесків в затримку мережі.

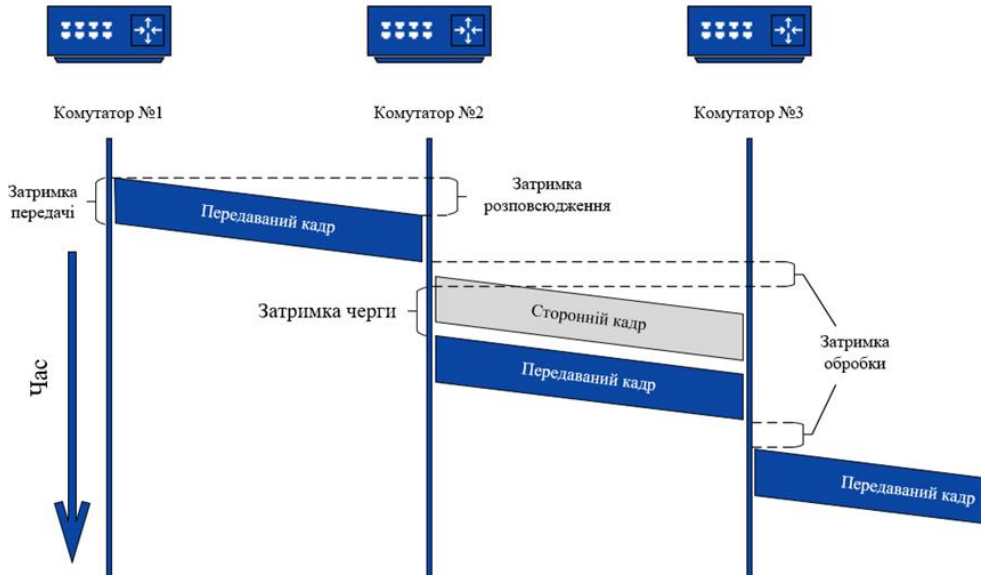


Рисунок 2.4 – Діаграма внесків в затримку мережі

Затримка передачі (Transmission Delay) – визначає, скільки часу потрібно для передачі всіх біт даних через фізичний канал зв'язку. Цей час прямо пропорційний розміру пакету та обернено пропорційний пропускній здатності каналу. Чим більший пакет або менша пропускна здатність каналу, тим довше триватиме передавання.

Затримка розповсюдження, або поширення (Propagation Delay) – це час, який потрібен сигналу, щоб пройти від відправника до отримувача. Він залежить від відстані між пристроями і швидкості поширення сигналу (наприклад, швидкості світла в оптичному волокні або кабелі). Довші відстані або повільніші носії сигналів збільшують час поширення.

Затримка обробки (Processing Delay) – визначає, скільки часу потрібно для обробки пакету в мережевих пристроях, таких як маршрутизатори, комутатори або

сервери. Цей час включає перевірку заголовків пакету, маршрутизацію та інші операції. Чим складніші операції обробки, тим більше часу потрібно.

Затримка черги (Queuing Delay) – це час, який пакет очікує в черзі перед тим, як бути переданим через мережу. Цей час залежить від навантаження на мережевий вузол і рівня трафіку. В умовах високого навантаження пакети можуть затримуватися довше, чекаючи своєї черги на передавання.

Затримка обробки на кінцевих пристроях (End-Device Processing Delay): Час обробки на кінцевих пристроях визначає, скільки часу потрібно для обробки даних на кінцевих пристроях перед передачею чи після отримання даних. Це залежить від продуктивності кінцевих пристроїв і типу додатка. Наприклад, обробка великих об'ємів даних на повільному пристрої може зайняти більше часу.

Тремтіння, джиттер (Jitter): Цей параметр характеризує коливання тривалості затримки між окремими пакетами [21]. Наприклад, якщо один пакет досягає кінцевого пункту за 30 мс, а наступний – за 40 мс, джиттер у цьому випадку становитиме 10 мс. Джиттер є критичним параметром для мультимедійних сервісів, де постійний потік даних забезпечує високу якість звуку чи зображення.

Для зменшення впливу тремтіння широко використовується буфер джиттеру Принцип роботи буфера джиттеру полягає у тимчасовому зберіганні вхідних пакетів, які надходять із різними затримками, та синхронізації їх перед відтворенням. Це дозволяє уникнути пропусків або спотворень у передачі голосу, відео чи інших потокових даних. Наприклад, у випадку, якщо пакет надходить із запізненням, буфер може компенсувати затримку, подаючи попередньо збережені дані, забезпечуючи таким чином безперервність потоку, як це зображено на рисунку 2.5 [22].

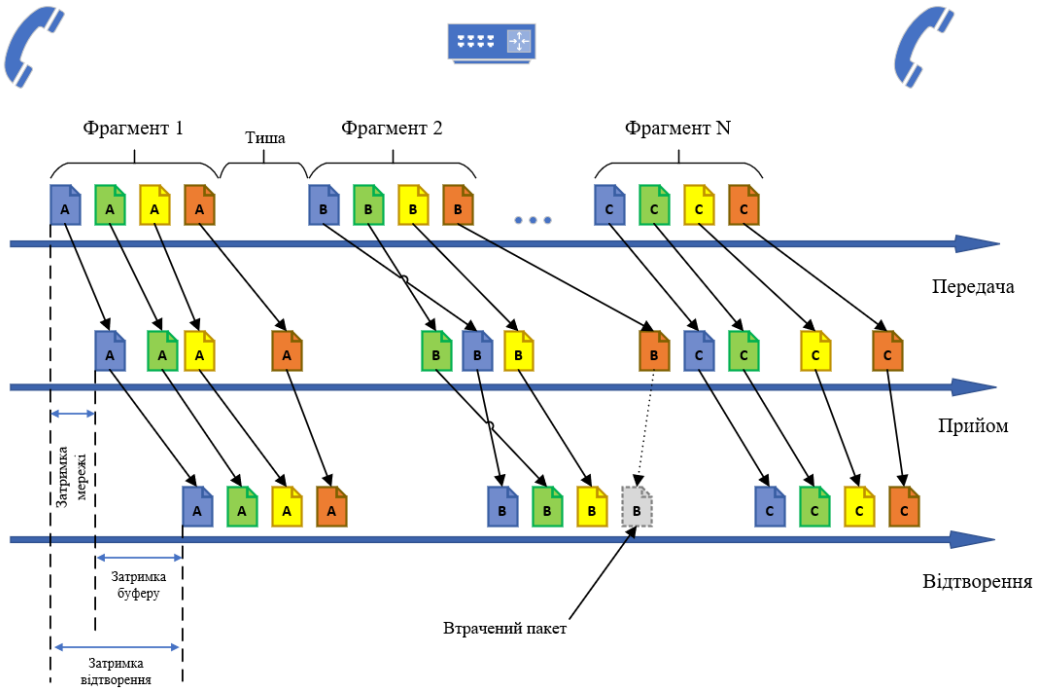


Рисунок 2.5 – Процес передачі та відтворення пакетів у мережі

Визначення оптимального розміру цього буфера є ключовим завданням, адже неправильний вибір може призвести до збільшення затримки зв'язку, як це зображено на рисунку 2.6.

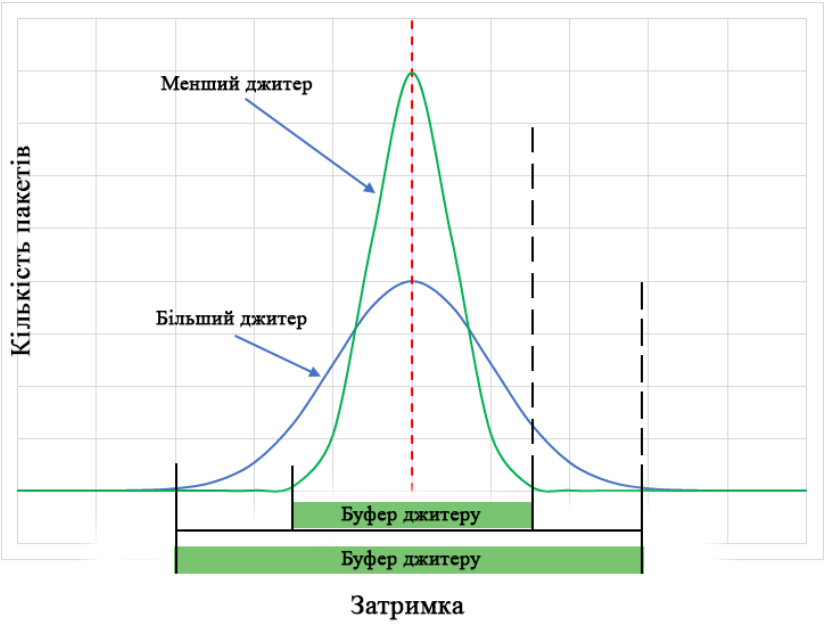


Рисунок 2.6 – Діаграма внесків в затримку мережі

Втрата пакетів (Loss): Під час передачі даних через мережу деякі пакети можуть бути втрачені, не досягши кінцевої точки. Втрата пакетів негативно впливає на якість даних, оскільки знижує цілісність переданої інформації. Оптимізація QoS спрямована на мінімізацію втрат, щоб забезпечити безперебійну передачу даних.

2.4 Мережеві технології для забезпечення якості обслуговування

Забезпечення якості обслуговування (QoS) – це сукупність технологій та методів управління мережевим трафіком, спрямованих на забезпечення надійності, мінімізації затримок та стабільності передачі даних у мережі [23]. QoS відіграє критичну роль у сучасних корпоративних мережах, де працюють різні типи трафіку – від базових веб-запитів до складних сервісів реального часу, таких як IP-телефонія чи відеоконференції. Мережі, що впроваджують QoS, здатні забезпечити необхідні ресурси для кожного типу трафіку, підтримуючи ефективність, стабільність і високу якість обслуговування для користувачів.

Основні технології QoS:

– Класифікація і маркування трафіку. Для ефективного управління трафіком першим кроком є класифікація даних. Класифікація передбачає розподіл пакетів у категорії, залежно від їхньої важливості або типу послуг, що забезпечуються. Для маркування таких пакетів використовується протокол DSCP (Differentiated Services Code Point), що дозволяє ідентифікувати пакети різних типів. Кожен пакет отримує мітку, яка вказує на його пріоритет, що дає можливість мережевим пристроям (маршрутизаторам, комутаторам) коректно обробляти його. Класифікація і маркування є базовими елементами для роботи інших технологій QoS, адже вони дозволяють чітко ідентифікувати критично важливий трафік, як VoIP або відеоконференції.

– Пріорітизація трафіку – це метод, за допомогою якого окремі типи трафіку отримують певний пріоритет в обслуговуванні. Наприклад, пакети з високим пріоритетом (наприклад, голосовий або відеотрафік) можуть оброблятися першими, навіть якщо мережа завантажена. Це гарантує, що чутливий до затримок

трафік не відчуватиме перешкод через низькопріоритетні дані, як електронна пошта чи завантаження файлів. Зокрема, технологія WFQ (Weighted Fair Queueing) забезпечує можливість розподілу пропускної здатності між різними чергами, виділяючи більше ресурсів тим потокам, які мають більший пріоритет.

– Політики керування трафіком (Traffic Policing) – це механізм обмеження трафіку, шляхом відкидання пакетів, які перевищують задану швидкість передачі, як це подано на рисунку 2.7 [24]. Це корисно для забезпечення дотримання певних правил у межах угод про якість обслуговування (SLA). Якщо обсяг даних перевищує встановлений ліміт, надлишкові пакети або відкидаються, або позначаються як «низькопріоритетні». Policing може бути жорстким способом контролю над споживанням пропускної здатності, особливо в корпоративних мережах з обмеженими ресурсами.

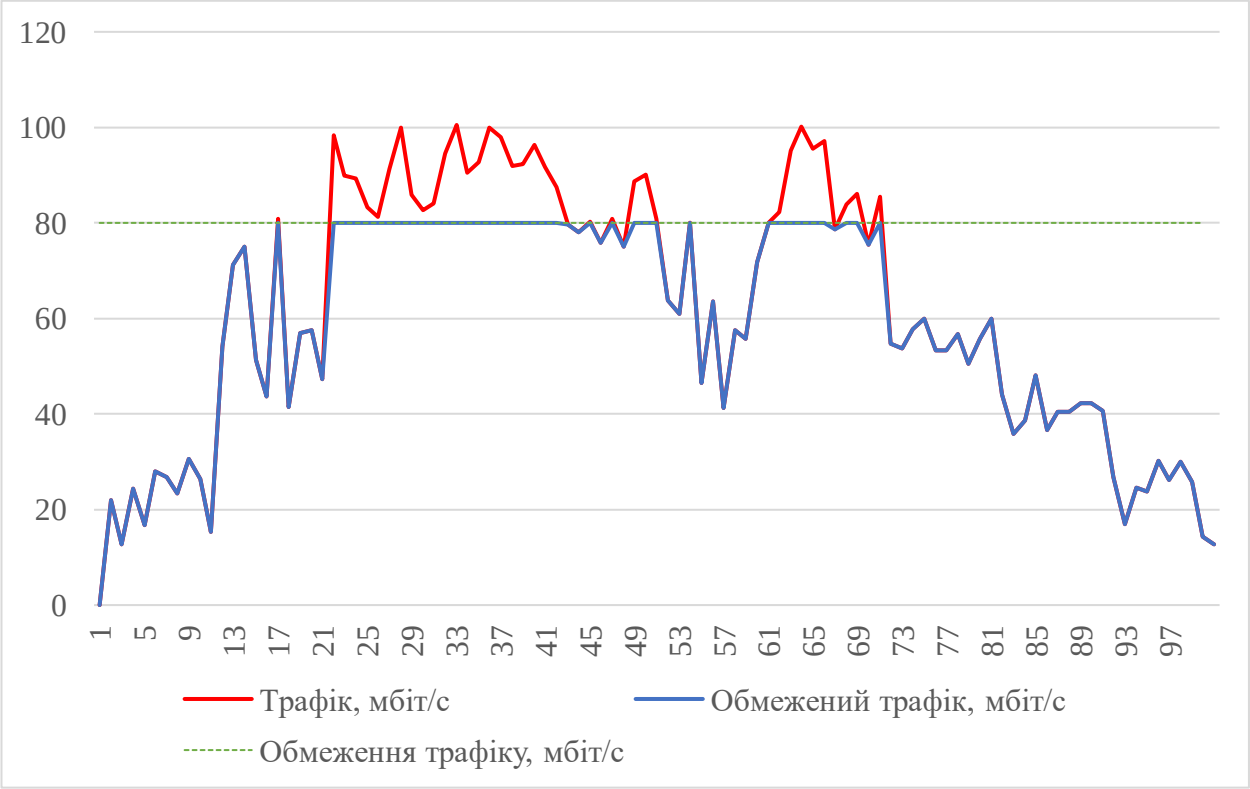


Рисунок 2.7 – Графік впливу політик керування трафіком на передачу інформації

– Формування (шейпінг) трафіку (Traffic Shaping). Мережеві пристрої працюють на фізичному рівні, передаючи біти зі швидкістю, визначеною тактовою частотою їх апаратного забезпечення.

З технічної точки зору, мережеві інженери можуть контролювати майже все, що стосується налаштування передачі даних, але вони не можуть сповільнити електричний або оптичний сигнал у кабелі довільно, за потребою. Тобто фізично зробити так, щоб сигнали передавалися повільніше, ніж дозволяє тактова частота пристрою неможливо з декількох причин, що пов’язані з принципами фізики сигналів, стандартами передачі даних, і технічними обмеженнями мережевих пристроїв:

– Фізична обмеженість лінії зв’язку та інтерфейсів: Мережеві інтерфейси, такі як Ethernet, оптичні канали або серійні порти, розроблені з конкретною частотою передачі даних. Ця частота – це результат не тільки електронної схеми пристрою, а й фізичних характеристик середовища передачі (наприклад, властивостей кабелю або волоконного світловоду). Ці фізичні параметри визначають максимальну частоту сигналу, яка може передаватися без значних втрат якості та цілісності сигналу.

– Протоколи та стандарти: Технології зв’язку, як Ethernet, мають чітко визначені стандарти швидкості передачі даних (наприклад, 10 Мбіт/с, 100 Мбіт/с, 1 Гбіт/с тощо). Кожен стандарт має свої технічні параметри, що стосуються частоти тактових імпульсів, кодування сигналів, електромагнітної сумісності тощо. Зміна тактової частоти означала б відступ від цього стандарту, що призвело б до порушення сумісності між пристроями.

– Частотна синхронізація між пристроями: Усі мережеві пристрої у зв’язку повинні бути синхронізовані, тобто «узгоджувати» частоту передачі для коректного обміну даними. Змінюючи частоту на одному з пристроїв, виникає порушення синхронізації, що спричинить втрати пакетів, збої у зв’язку, а іноді й повне припинення зв’язку.

Коли необхідно обмежити швидкість передачі даних (наприклад, до рівня меншого, ніж підтримує канал), пристрої використовують метод тимчасового

призупинення передачі – «формування» трафіку. Замість того, щоб зменшити швидкість передачі інформації протягом всього проміжку часу, мережевий пристрій чергує періоди активної передачі пакетів з паузами. Таким чином знижується середнє значення швидкості передачі інформації, не обмежуючи загальну пропускну здатність каналу що використовується для передачі даних.

Розглянемо конкретний приклад: припустимо, канал має пропускну здатність 64 кбіт/с. Якщо необхідно обмежити середню швидкість передачі даних між заданими вузлами до 32 кбіт/с, то для досягнення цієї мети мережевий пристрій повинен забезпечити періодичне передавання даних упродовж 50% загального часу, залишаючи інші 50% часу в режимі тиші, як це зображено на рисунку 2.8. Це дозволяє не лише знизити середню пропускну здатність до половини фізичної пропускної здатності каналу, тобто до 32 кбіт/с, але й звільнити ресурс каналу для передачі інших пакетів.

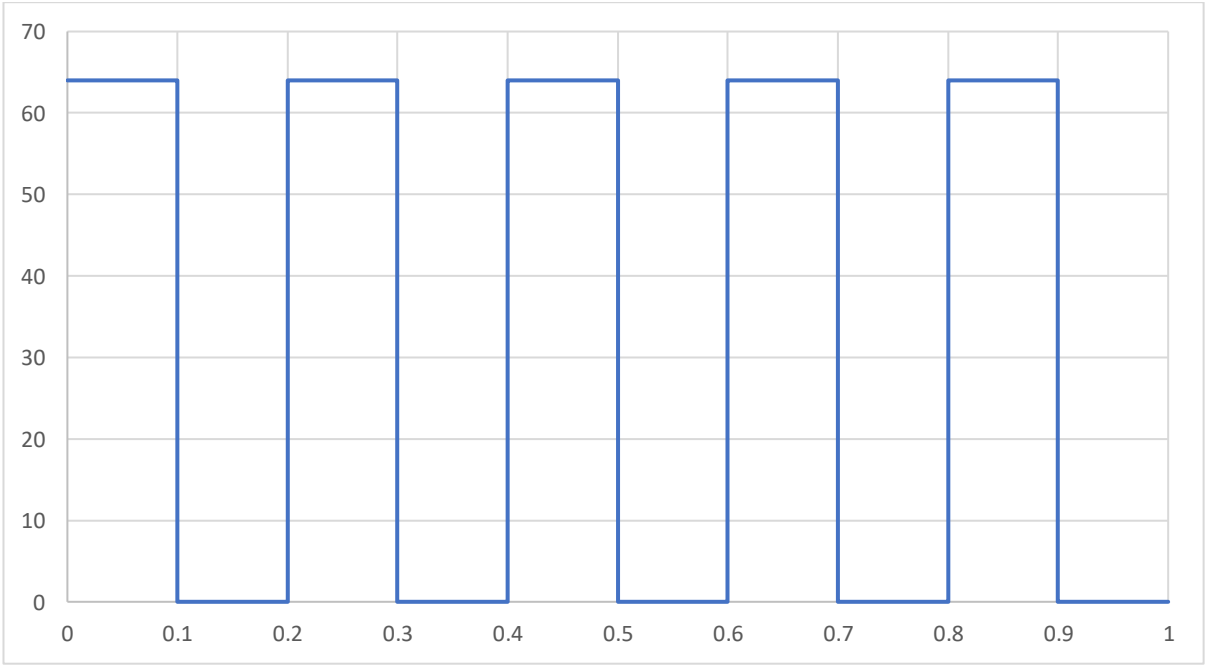


Рисунок 2.8 – Графік імпульсно-періодичної передачі даних з середньою швидкістю 32 кбіт/с

Якщо канал з пропускною здатністю 64 кбіт/с потрібно обмежити до середньої швидкості 28 кбіт/с, мережевий пристрій повинен передавати дані

протягом 44% часу та зупиняти передачу на 56% часу. Це досягається шляхом чергування активних періодів передачі даних і пауз, забезпечуючи таким чином зниження середньої пропускної здатності до бажаного рівня.

$$T_{\text{передачі}} = V/C = \frac{28}{64} = 0{,}4375 \approx 44\%,$$

де V – необхідна швидкість передачі інформації, кбіт/с;

C – пропускна здатність каналу, кбіт/с;

28 / 64 = 0.75, тобто передача даних здійснюється 75% від загальної тактової частоти каналу, досягаючи таким чином необхідного середнього показника у 96 кбіт/с, як це подано на рисунку 2.9.

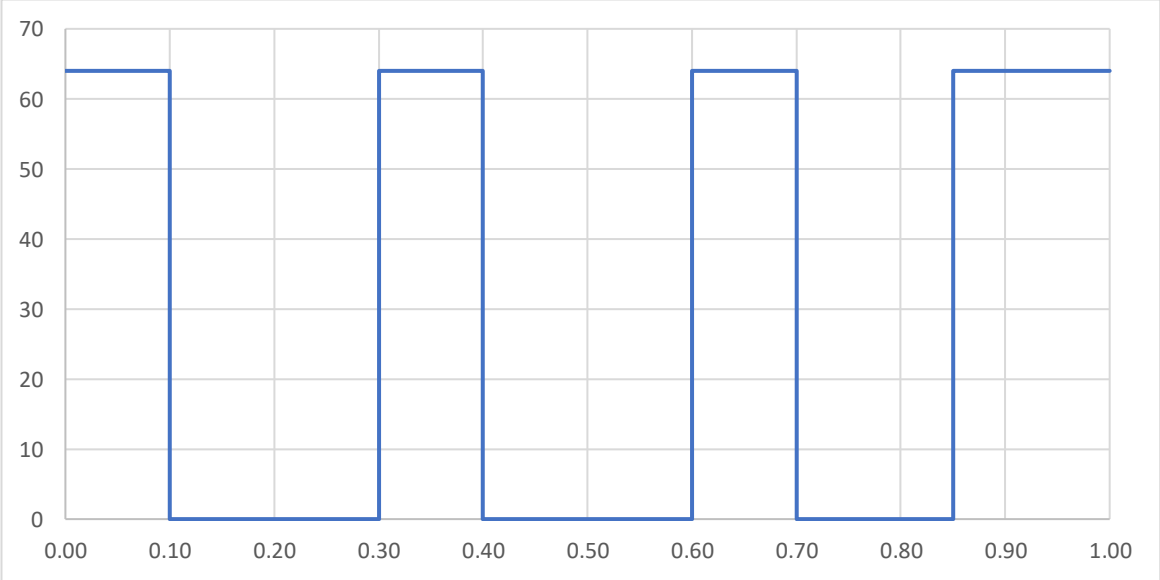


Рисунок 2.9 – Графік імпульсно-періодичної передачі даних з середньою швидкістю 28 кбіт/с

Формування є ключовою технологією в QoS, яка дозволяє обмежувати швидкість передачі певних типів даних у мережі для запобігання перевантаженню каналів. Traffic Shaping згладжує трафік, розподіляючи його протягом часу, як це зображено на рисунку 2.10 [24]. Це дозволяє уникати перевантажень мережевої

інфраструктури. Зазвичай застосовується для нестабільного трафіку, наприклад відео або файлових передач, які можуть переривати інші, важливі для бізнесу сервіси. Шейпінг реалізується через буферизацію пакетів та обмеження їх виходу на мережевий канал.

Існують дві причини використовувати формування трафіку:

– Задля уникнення відкидання трафіка політиками провайдера, можливо сформувати трафік таким чином, щоб не перевищувати ліміт гарантованої смуги пропускання (CIR).

– Для запобігання блокування виходів. При переході від більш швидкісного інтерфейсу до менш швидкісного/більш навантаженого інтерфейсу, можуть виникнути втрати у вихідній черзі. В цій ситуації також варто використати формування, щоб гарантувати, що всі дані будуть доставлені.

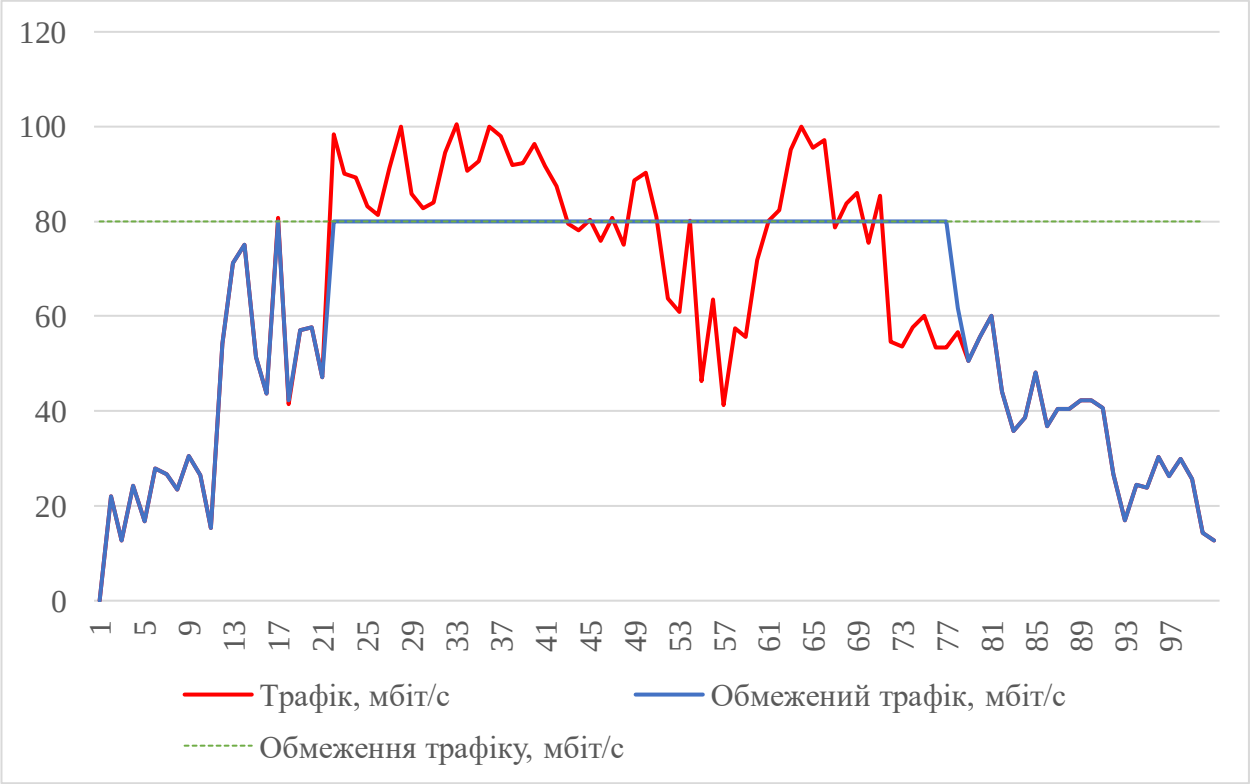


Рисунок 2.10 – Графік впливу формування трафіку на передачу інформації

Управління чергами (Queue Management). Технології управління чергами дозволяють налаштовувати порядок обробки трафіку в разі перевантаження. Один

із поширених методів – це зважена випадкова черга раннього виявлення (Weighted Random Early Detection, WRED), яка автоматично регулює чергу, відкидаючи менше важливі пакети під час завантаження [25]. Це дозволяє уникнути заповнення буферів мережевих пристроїв і сприяє стабільній роботі мережі. Queue Management також включає такі методи, як First-In-First-Out (FIFO) та Priority Queuing, де пакети обробляються в порядку прибуття або відповідно до пріоритетів, що зображено на рисунку 2.11.

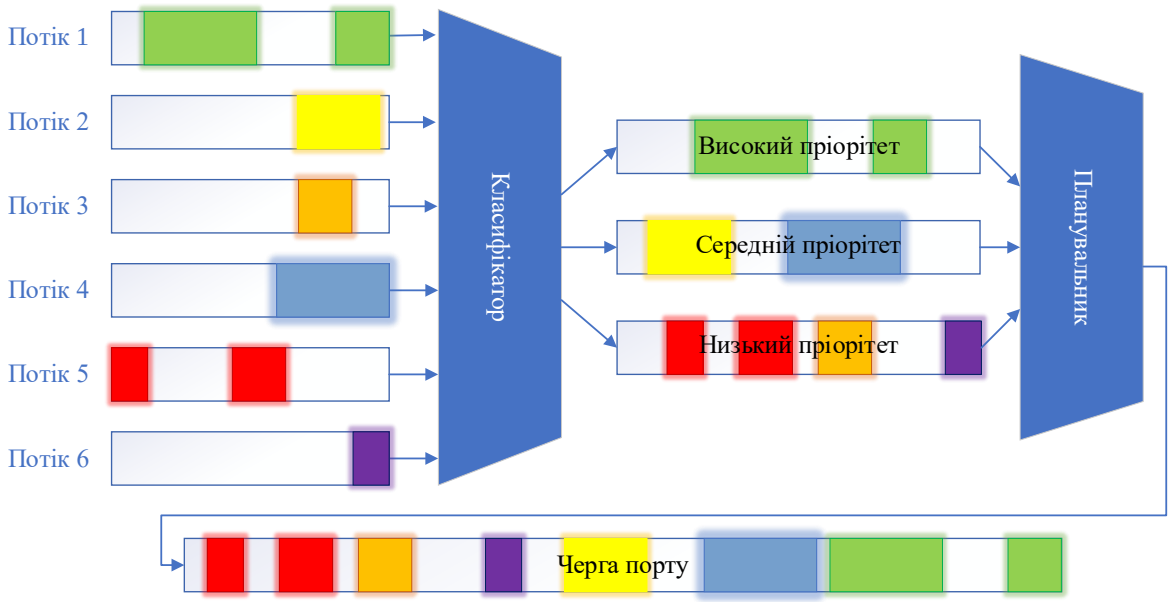


Рисунок 2.11 – Чергування за пріоритетом

Резервування пропускної здатності (Bandwidth Reservation). Цей метод передбачає попереднє виділення певної пропускної здатності для конкретних видів трафіку. У мережах із високою завантаженістю Bandwidth Reservation дозволяє забезпечити мінімальну затримку для критично важливих додатків, таких як IP-телефонія чи відеоконференції. Це особливо корисно у великих мережах, де є потреба в гарантованій пропускній здатності для забезпечення якості обслуговування.

Сучасні мережі поєднують різні методи QoS, залежно від специфіки трафіку та вимог підприємства. У корпоративних мережах, де одночасно обробляється велика кількість даних, ефективне поєднання класифікації, пріоритизації та

резервування ресурсів дозволяє уникнути затримок і забезпечує високу якість обслуговування. Наприклад, для IP-телефонії може бути виділений окремий сегмент пропускної здатності з високим пріоритетом, щоб уникнути затримок у голосових дзвінках, а для звичайного трафіку можуть бути застосовані методи шейпінгу та технології застосування політик. Це дозволяє адаптувати мережу до специфічних вимог кожного типу трафіку.

Забезпечення QoS у мережах дозволяє підвищити надійність і передбачуваність обслуговування, що особливо важливо для критично важливих додатків і сервісів. Використання QoS допомагає підприємствам досягти високої ефективності використання ресурсів і підтримувати стабільний рівень обслуговування в умовах змінної мережевої активності. Водночас впровадження QoS може вимагати значних витрат на модернізацію обладнання та налаштування, що може ускладнити адаптацію у великих мережах.

QoS є незамінним елементом управління трафіком у сучасних мережах. Його інтеграція та поєднання з іншими технологіями, як-от MPLS та SD-WAN, дозволяють гнучко управляти мережею й оптимізувати її для надання стабільних і надійних послуг користувачам.

3 СИНТЕЗ КОМП'ЮТЕРНОЇ СИСТЕМИ

3.1 Цілі впровадження комп'ютерної системи

Основною метою впровадження системи є організація доступу до інструментів і ресурсів для роботи з даними, підвищення ефективності та безпеки діяльності підприємства завдяки інтеграції сучасних технологій відеоспостереження та IP-телефонії.

Система призначена організувати ефективну, надійну та безпечну передачу даних, забезпечити безперебійний зв'язок за допомогою IP-телефонії, а також здійснювати високоякісний відеомоніторинг для підвищення рівня безпеки і контролю на підприємстві.

3.2 Вибір і обґрунтування принципів побудови системи

Для побудови комп'ютерної системи підприємства визначено ключові принципи, які мають забезпечити її ефективність, надійність і безпеку:

– Модульність. Система має бути побудована за модульним принципом, що дозволить у майбутньому легко оновлювати, масштабувати та замінювати окремі компоненти (наприклад, відеокамери чи сервери) без впливу на загальну структуру. Це забезпечить гнучкість та адаптивність до можливих змін чи інтеграції нових технологій.

– Доступність. Для забезпечення безперервної роботи системи мають бути впроваджені резервування та надмірність, що гарантуватимуть відмовостійкість.

– Безпека. Система має бути оснащена механізмами шифрування, аутентифікації та контролю доступу для захисту інформаційних потоків. Це має мінімізувати ризики несанкціонованого доступу та забезпечити конфіденційність переданих даних.

– Масштабованість. Завдяки модульному дизайну та резервуванню система має бути готова до розширення, враховуючи можливе зростання обсягів даних чи збільшення кількості користувачів.

Дотримання вищенаведених принципів забезпечить створення гнучкої, безпечної та ефективної системи, яка відповідатиме потребам підприємства.

3.3 Розробка схеми функціональної структури

Функціональна модель підприємства, що наведена на рисунку 3.1 визначає основні завдання кожного підрозділу, що забезпечують ефективність бізнес-процесів.

Метою розробки є створення моделі, яка відображає зв'язки між підрозділами ДП «Антонов», їх основні функції та використання комп'ютерних систем для підтримки бізнес-процесів.

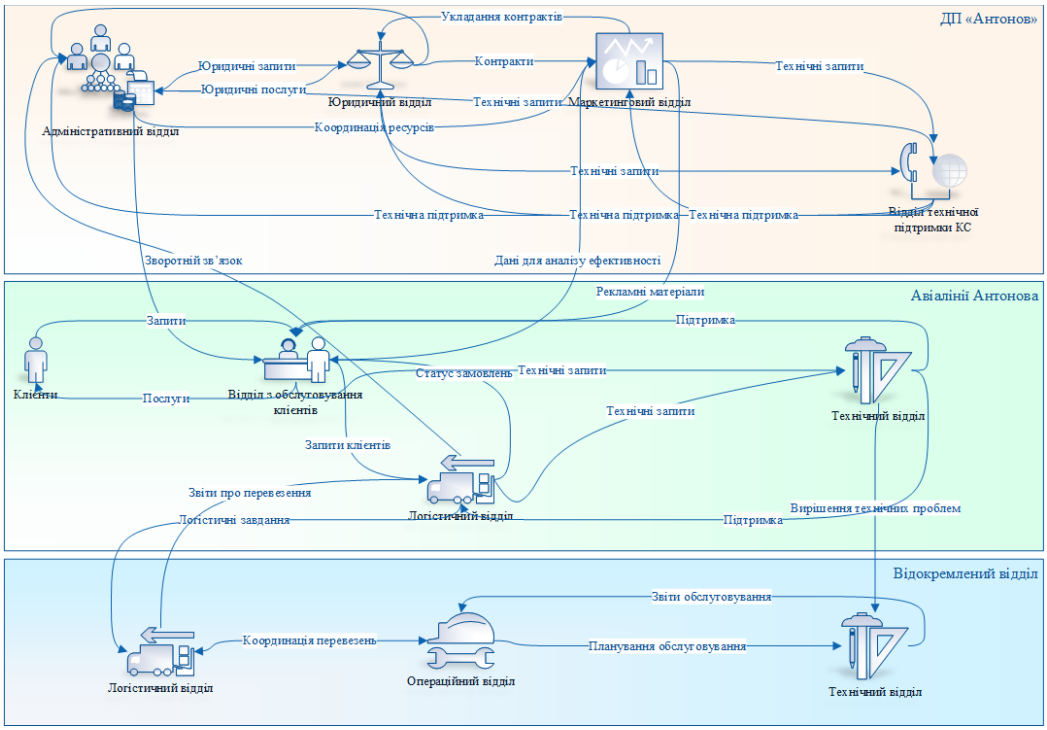


Рисунок 3.1 – Схема функціональної структури підприємства

Мережеві завдання підприємства визначаються функціональними потребами підрозділів. Для кожного з них важливе ефективне взаємодія, обмін інформацією та доступ до ресурсів.

Адміністративний відділ потребує стабільної мережевої інфраструктури для координації між підрозділами через корпоративну IP-телефонію та

відеоконференції. Доступ до систем управління документами (DMS) оптимізує документообіг, а фінансові програми забезпечують прозорість бізнесу.

Юридичний відділ використовує мережеві сервіси для обробки контрактів і доступу до баз даних, зберігаючи конфіденційність документів через захищені канали зв’язку.

Маркетинговий відділ застосовує хмарні сервіси для аналізу ефективності рекламних кампаній, а також IP-телефонію для взаємодії з клієнтами. Тісна інтеграція з логістичним відділом сприяє реалізації маркетингових стратегій.

Відділ технічної підтримки займається моніторингом мережі, управлінням відеонаглядом і IP-телефонією, а також захистом даних через резервне копіювання та інформаційну безпеку.

Відділ обслуговування клієнтів Авіаліній Антонова використовують IP-телефонію та веб-платформи для комунікації з клієнтами, а інтеграція з логістичним відділом дозволяє ефективно обробляти запити. Логістичний відділ організовує обслуговування літаків через спеціалізовані програми та захищені канали зв’язку з міжнародними партнерами.

Технічний відділ авіаліній контролює технічний стан літаків через мережеві системи, оптимізуючи ремонт і діагностику.

Підрозділ у Лейпцигу координує логістичну діяльність за допомогою хмарних сервісів та інтеграції з головним офісом для обміну даними про вантажопотоки. Операційний відділ аеропорту передає технічні запити через захищені канали, забезпечуючи швидке вирішення проблем.

Всі підрозділи інтегровані в єдину корпоративну мережу для забезпечення ефективної роботи та збереження конфіденційності інформації через захищені канали зв’язку.

Схема функціональної структури мережі, що наведена на рисунку 3.2 відображає взаємодію основних відділів підприємства та їх мережеві ресурси, що забезпечують ефективне функціонування організації.

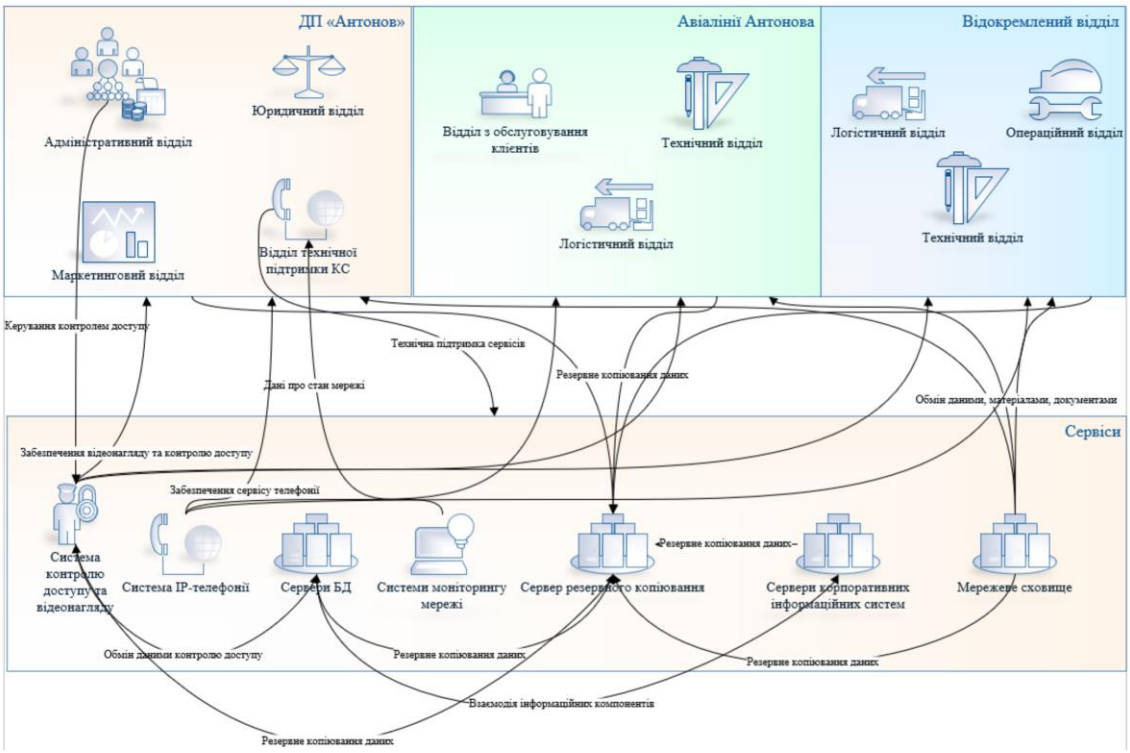


Рисунок 3.2 – Схема функціональної структури мережі

3.4 Формулювання технічних вимог до комп’ютерної системи

3.4.1 Вимоги до структурних характеристик і режимів функціонування системи

Система має реалізовувати наступний функціонал:

– Система має забезпечити можливість авторизованого доступу до мережевих ресурсів для співробітників та підключення пристроїв у межах підприємства.

– Корпоративна мережа має забезпечити обробку та передачу голосового трафіку для внутрішніх дзвінків.

– Система має підтримувати трансляцію та запис відеопотоків із камер відеоспостереження в реальному часі з можливістю зміни параметрів запису залежно від наявності руху.

– Мережа має забезпечити автоматичне збереження відеозаписів у централізованому сховищі з доступом до архіву за часовими мітками.

– Корпоративна мережа має забезпечити автоматичне резервне копіювання даних, включаючи відеозаписи, конфігурації систем і користувацькі файли сховище з можливістю відновлення.

Передача даних між підсистемами в умовах територіальної роз'єднаності повинна здійснюватися із використанням захищених каналів зв'язку (VPN).

У системі відеонагляду для ефективного використання пропускної здатності застосовувати технологію VBR (Variable Bitrate), яка автоматично регулює бітрейт в залежності від складності зображення на відео.

Комп'ютерна мережа повинна охоплювати наступні структурні підрозділи підприємства:

– ДП «Антонов»: Адміністративний відділ, Юридичний відділ, Маркетинговий відділ, Відділ технічної підтримки комп'ютерних систем, Серверне приміщення.

– Авіалінії Антонова: Логістичний відділ, Технічний відділ, Відділ обслуговування клієнтів.

– Відокремлений підрозділ авіакомпанії «Авіалінії Антонова»: Логістичний відділ, Технічний відділ, Операційний відділ, Відділ технічної підтримки комп'ютерних систем.

Розташування відділів підприємства наведено на плані, представленому на рисунку 3.3.

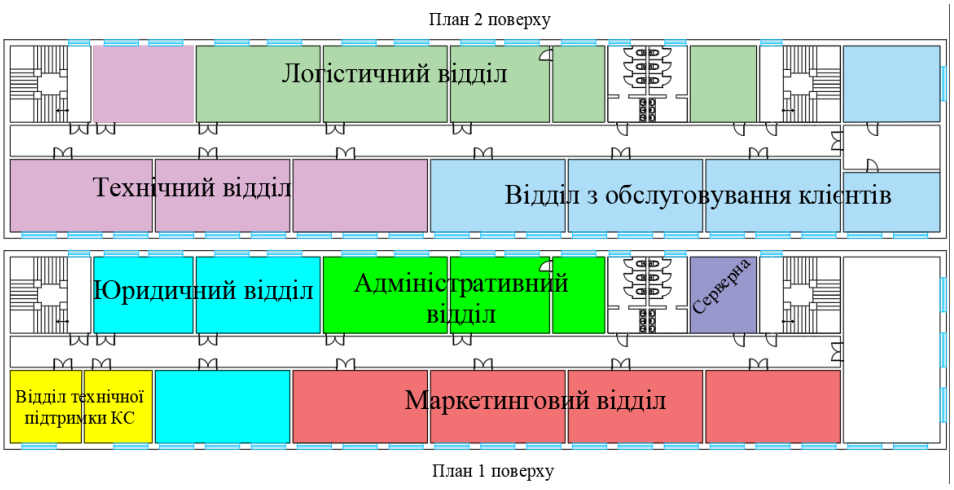


Рисунок 3.3 – План розташування відділів підприємства

3.4.2 Вимоги до надійності

Система повинна забезпечувати високий рівень надійності, що передбачає доступність не менше 99,9% часу протягом року, резервування ключових компонентів для уникнення простоїв, а також можливість повного відновлення роботи протягом години у разі аварії. Цілісність даних реалізується за допомогою механізму відмовостійкості системи RAID 6 та резервного копіювання з можливістю відновлення за останні 24 години.

3.4.3 Вимоги до розвитку системи

Система повинна бути розроблена з урахуванням можливостей подальшого розширення та модернізації. Вона має забезпечувати збільшення кількості хостів та обслуговуваного трафіку на 30% без суттєвих змін у її структурі.

3.4.4 Вимоги до системи захисту інформації, спрямовані на запобігання несанкціонованому доступу

Система повинна бути розроблена з застосуванням двофакторної аутентифікації для доступу до критичних компонентів системи за допомогою одноразових кодів (Time-based One-Time Password) та використанням ролей, політик доступу, ACL списків для обмеження доступу лише до необхідних даних, ресурсів і функцій.

3.4.5 Вимоги до фізичної безпеки доступу

Доступ до будівлі обмежується за допомогою турнікетів, які встановлюються на всіх основних входах. Турнікети оснащуються зчитувачами карт доступу. Несанкціонований доступ заборонено, доступ дозволяється лише з відповідними правами доступу, які визначаються через корпоративну систему контролю доступу. Турнікети автоматично реєструють час входу та виходу, що дозволяє забезпечити контроль за присутністю персоналу та підвищує рівень безпеки будівлі.

Доступ до серверного приміщення дозволяється тільки персоналу з відповідним рівнем доступу. Для доступу до приміщення використовуються біометричні системи аутентифікації.

Встановлення датчиків руху та сенсорів відкриття дверей для реагування на спроби несанкціонованого доступу.

3.4.6 Вимоги до ергономіки

Робочі столи та крісла для операторів ПК повинні бути розташовані так, щоб забезпечити природну та зручну позу під час роботи. Це передбачає правильну підтримку спини, рук і ніг, а також оптимальне розташування клавіатури, миші та монітора. Робочі місця мають бути обладнані системами регулювання для адаптації до різних фізичних параметрів та потреб користувачів, включаючи налаштування висоти столу й крісла, нахилу спинки, а також нахилу та висоти монітора. Відповідно до пункту 2.3 Державних санітарних правил і норм роботи 3.3.2.007-98, площа одного робочого місця оператора ПК повинна становити не менше 6,0 м² [26].

3.4.7 Вимоги до показників призначення

Система повинна забезпечувати підтримку роботи робочих станцій кількістю до 137 одиниць, до 80 одночасних голосових дзвінків при використанні кодеку g711. Згідно рекомендацій ITU G.114, значення затримки в один бік не має перевищувати 150 мс [27].

Система повинна забезпечувати безперервне відеоспостереження 12-ма камерами у режимі 24/7 з можливістю зберігання записів за останні три місяці. Відеоспостереження має функціонувати в режимах:

– За відсутності руху з роздільною здатністю 1280x720 пікселів, кадровою частотою 15 кадрів/с та бітовою швидкістю до 1 Мбіт/с.

– При виявленні руху з роздільною здатністю 1280x720 пікселів, кадровою частотою 30 кадрів/с та бітовою швидкістю до 3 Мбіт/с.

3.4.7.1 Розрахунок навантаження на комп’ютерну систему

Для стабільної роботи комп’ютерної системи важливо врахувати мережеве навантаження, яке створюють пристрої. Необхідно визначити мінімальну пропускну здатність для якісного голосового зв’язку без затримок і втрат при максимальному завантаженні.

Крім корисного трафіку, значну частку займають накладні витрати мережевих протоколів (заголовки), що впливають на загальний обсяг трафіку. Їх врахування є критичним під час планування пропускної здатності.

В даному випадку використовується кодек G.711, що генерує потік даних у 64 кбіт/с [28]. Розрахунок мережевого трафіку базується на розмірах голосового навантаження та структурі заголовків протоколів RTP, UDP, IP та Ethernet. Деталізуємо цей розрахунок для визначення загальної пропускної здатності, необхідної для обробки 80 одночасних дзвінків.

Таблиця 3.1 – Вхідні дані

Протокол	Розмір заголовку, байт
Real-Time Transport Protocol (RTP)	12
User Datagram Protocol (UDP)	8
Internet Protocol (IP)	20
Ethernet	14+4 (Контрольна сума CRC)
Загалом:	58

Розмір корисного навантаження визначається на основі швидкості кодека та тривалості аудіофрагмента, що кодується в одному пакеті. Наприклад, якщо кодек G.711 створює потік даних зі швидкістю 64 кбіт/с, це означає, що за одну секунду він генерує 64 000 біт інформації.

Тривалість одного аудіофрагмента (час, що кодується в одному пакеті) становить 20 мілісекунд, тобто 1/50 секунди, отже:

$$64\ \text{кбіт/с} * 20\ \text{мс} = 64\ \text{кбіт/с} * 0.02\ \text{с} = 1{,}28\ \text{кбіт} = 160\ \text{байт}$$

Розмір одного пакета включає корисне навантаження та заголовки, які додаються до кожного пакету для забезпечення його правильної маршрутизації та управління передачею:

$$160 \text{ байт} + 58 \text{ байт} = 218 \text{ байт}$$

Для обчислення трафік голосового каналу необхідно помножити розмір одного пакета на кількість пакетів, що передаються в секунду:

$$218 \text{ байт} * 50 \text{ пакетів/с} = 10900 \text{ байт/с} = 87200 \text{ біт/с} = 87{,}2 \text{ кбіт/с}$$

Оскільки трафік йде в обох напрямах (вхідний і вихідний), двосторонній трафік становить:

$$87{,}2 \text{ кбіт/с} * 2 = 174{,}4 \text{ кбіт/с}$$

Для 80 одночасних дзвінків загальний трафік:

$$174{,}4 \text{ кбіт/с} * 80 = 13952 \text{ кбіт/с} = 13{,}952 \text{ Мбіт/с}$$

Задля більш прогнозованого підходу в розрахунках навантаження мережі системою відеонагляду, за вхідні параметри взято найважчий для мережі сценарій. У цьому випадку передбачається, що всі 12 камер працюють в сталому режимі з роздільною здатністю 1280x720 точок, бітрейтом 3 Мбіт/с та частотою кадрів 30 кадрів/с. Це дозволяє отримати більш точну оцінку пропускної здатності мережі, враховуючи сценарій з високим навантаженням на мережу.

Розмір корисного навантаження дорівнює значенню бітрейту що дорівнює 3 Мбіт/с.

$$3\ \text{Мбіт/с} = 375000\ \text{байт/с}$$

Корисне навантаження в одному пакеті при максимальному розміру блоку Ethernet (MTU), що дорівнює 1500 байтів [29]:

$$1500 - 58 = 1442\ \text{байт}$$

Кількість пакетів на секунду для однієї камери:

$$\text{Кількість пакетів/с} = \frac{375000}{1442} = 260\ \text{пакетів/с}$$

Загальний трафік на одну камеру (з урахуванням заголовків):

$$260 * 1500\ \text{байт} = 390000\ \text{байт/с} = 3{,}12\ \text{Мбіт/с}$$

Загальний трафік для 12 камер:

$$3{,}12 * 12 = 37{,}44\ \text{Мбіт/с}$$

Таким чином, для 12 камер загальна пропускна здатність для відеонагляду складе 37,44 Мбіт/с. Цей розрахунок базується на прогнозованому сценарії, де всі камери одночасно передають відео з максимальним бітрейтом.

Для розрахунку швидкості мережі, необхідної для обслуговування хостів взято статистику з робочих станцій та обрано найбільш активний з наявних, тому його трафік розглядається як такий, що відображає пікове навантаження для одного пристрою. Це дає змогу спрогнозувати максимальні навантаження в мережі. Статистику мережевої активності наведено на рисунку 3.2.

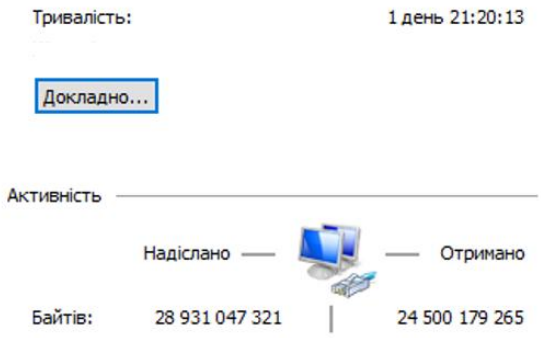


Рисунок 3.4 – Статистика найактивнішої робочої станції

Перш за все, треба визначити загальний обсяг трафіку, який був переданий через мережу для одного хоста. Для цього додаємо кількість байт, що були надіслані, до кількості байт, що були отримані та переведено у біт:

$$28\,931\,047\,321 \text{ байт} + 24\,500\,179\,265 \text{ байт} = 53\,431\,226\,586 \text{ байт}$$
$$53\,431\,226\,586 \text{ байт} * 8 \text{ біт} = 427\,449\,812\,688 \text{ біт} \approx 427\,450 \text{ Мбіт}$$

Маємо тривалість збору статистики: 1 день, 21 година, 20 хвилин, 13 секунд. Необхідно перевести час, за який цей трафік був переданий, в секунди.

$$1 \text{ день} = 24 \text{ год.} = 24 \text{ год.} * 3600 \text{ с.} = 86400 \text{ с.}$$
$$21 \text{ год.} * 3600 \text{ с.} = 75600 \text{ с.}$$
$$20 \text{ хв.} = 20 \text{ хв.} * 60 \text{ с.} = 1200 \text{ с.}$$
$$86400 \text{ с.} + 75600 \text{ с.} + 1200\text{с.} + 13\text{с.} = 163213 \text{ с.}$$

Обчислення середньої статистичної пропускної здатності одного хоста:

$$\frac{427\,450 \text{ Мбіт}}{163213 \text{ с.}} \approx 2{,}62 \text{ Мбіт/с}$$

Сумарний трафік для кінцевих пристроїв розраховується шляхом добутку трафіку на максимальну кількість вузлів:

$$2{,}62\,\text{Мбіт/с} * 137\ \text{од.} \approx 359\,\text{Мбіт/с}$$

Загальна швидкість:

$$359\,\text{Мбіт/с} + 37{,}44\,\text{Мбіт/с} + 13{,}952\,\text{Мбіт/с} = \ 410.392\,\text{Мбіт/с}$$

Виходячи з розрахунків для забезпечення стабільної роботи мережі з 137 хостами, системою IP-телефонії (до 80 одночасних дзвінків) і відеонаглядом (до 12 камер із роздільною здатністю 1280x720 точок, кадровою частотою 30 кадрів/с та бітрейтом 3 Мбіт/с), пропускна здатність мережі має бути не менше 410.392 Мбіт/с.

Отже, пропускна здатність на рівні доступу для робочих станцій має становити 100 Мбіт/с, для серверів 1 Гбіт/с.

3.4.7.2 Розрахунок об’єму дискового масиву системи відеонагляду

Згідно вимог до показників призначення система відеоспостереження повинна забезпечувати безперервний запис відеопотоків 24/7 із можливістю зберігання архіву записів за останні 3 місяці, та відповідно вимогам до надійності використовувати механізм RAID 6.

Для забезпечення безперервного запису відеопотоків з IP-камер протягом заданого періоду необхідно врахувати такі фактори, як бітрейт камер, тривалість зберігання архіву, обсяг доступного місця в дисковому масиві та тип конфігурації RAID, який використовується для захисту даних.

Камери генерують дані з максимальною швидкістю 37,44 Мбіт/с. Здійснено переведення Мбіт/с у Мбайт/с:

$$\frac{37{,}44\,\text{Мбіт/с}}{8\ \text{біт}} = 4{,}64\,\text{Мбайт/с}$$

Один місяць прийнято за 30 діб, здійснено переведення 3-х місяців у секунди.

$$30 \text{ діб} * 3 \text{ міс.} * 24 \text{ год.} * 3\,600 \text{ сек.} = 7\,776\,000 \text{ сек.}$$

Розраховано загальний об’єм даних створюваний системою відеонагляду протягом 3-х місяців. Отриманий результат переведено в ТБайт.

$$4{,}64 \text{ Мбайт/с} * 7\,776\,000 \text{ сек.} = 36\,391\,680 \text{ Мбайт}$$

$$\frac{36\,391\,680 \text{ Мбайт}}{1024 * 1024} = 34.71 \text{ ТБайт}$$

Для забезпечення надмірності при можливому зростанні обсягу даних і навантаження, відповідно до вимог розвитку системи, до розрахованого об’єму відеоданих додано 30% резерву на потенційне розширення системи. Згідно рекомендацій для забезпечення обслуговування жорстких дисків (дефрагментації) необхідно додатково резервувати 15% від об’єму диску [30].

$$(34.71 \text{ ТБайт} + 30\%) + 15\% = 51.89 \text{ ТБайт}$$

Для визначення мінімально необхідного об'єму одного диска в масиві RAID 6, використано формулу $S * (N - 2)$, де N – кількість дисків в масиві, S – об'єм найменшого диску. Перетворивши цю формулу, отримано рівняння для визначення мінімального об'єму одного диска.

$$\frac{51.89 \text{ ТБайт}}{12 - 2} = 5.18 \text{ ТБайт}$$

Згідно результату кожен диск повинен мати ємність не менше 5,18 ТБ. Оскільки на ринку представлені накопичувачі з ємністю 6 ТБ, вони відповідатимуть вимогам і будуть використані для забезпечення необхідної надмірності.

3.5 Оцінка навантаження на мережу та надмірності пропускної здатності

Для забезпечення оптимального функціонування локальної мережі важливо точно оцінити інтенсивність вихідного трафіку та максимальне навантаження. Розрахунки базуються на ключових параметрах мережі, таких як пропускна здатність каналу, швидкість генерації трафіку та середня довжина повідомлень.

Для хостів обрано максимальну довжину кадру Ethernet, що становить 1500 байт [29], оскільки це дозволить передбачити найважчий сценарій, оскільки об’ємніші кадри створюватимуть більше затримок у мережі. Це дозволяє провести аналіз, враховуючи можливі пікові навантаження.

$$l = \frac{1\,500\ \text{байт} + 1\,500\ \text{байт} + 218\ \text{байт}}{3} \approx 1\,073\ \text{байт} = 8\,584\ \text{біт}$$

Об’єм трафіку:

$$S = 410.392\ \text{Мбіт/с} = 410\,392\,000\ \text{біт/с}$$

Пропускна здатність мережі:

$$C = 1\ \text{Гбіт/с} = 1\,000\,000\,000\ \text{біт/с}$$

Розрахунок інтенсивності обслуговування та інтенсивності трафіку:

$$\mu_{\text{вих}} = \frac{C}{l} = \frac{1\,000\,000\,000\ \text{біт/с}}{8\,584\ \text{біт}} = 116\,495{,}81\ \text{пакетів/с}$$

$$\lambda = \frac{S}{l} = \frac{410\,392\,000\ \text{біт/с}}{8\,584\ \text{біт}} = 47\,808{,}95 \approx 47\,809\ \text{пакетів/с}$$

Розрахунок коефіцієнту затримки на рівні розподілу:

$$\rho = \frac{\lambda}{\mu_{\text{вих}}} = \frac{47\,809\,\text{пакетів/с}}{116\,495{,}81\,\text{пакетів/с}} = 0{,}4104$$

Розрахунок коефіцієнту зайнятості комутатора:

$$r = \frac{\rho}{1-\rho} = \frac{0{,}4104}{1-0{,}4104} = 0{,}6960$$

Розрахунок середньої довжини черги в системі, яка базується на M/M/1 моделі:

$$L_{\text{черги}} = \frac{\rho^2}{1-\rho} = \frac{0{,}4104^2}{1-0{,}4104} = \frac{0{,}1681}{0{,}590} \approx 0{,}285$$

Значення $L_{\text{черги}}$ менше за одиницю свідчить про те, що мережа здатна ефективно обробляти трафік, забезпечуючи мінімальні затримки та високу продуктивність.

Розрахунок середньої затримки кадру:

$$\text{T} = \frac{1}{\mu_{\text{вих}} - \lambda} = \frac{1}{116\,495{,}81 - 47\,808{,}95} = 1{,}456 * 10^{-5}\ \text{с} = 14{,}56\ \text{мкс}$$

Розрахунок середнього часу перебування пакета в черзі:

$$T_{\text{очік}} = \frac{L_{\text{черги}}}{\lambda} = \frac{0{,}285}{47\,808{,}95} = 5{,}96 * 10^{-6}\ \text{с} \approx 5{,}96\ \text{мкс}$$

Результати розрахунків демонструють, що мережа з пропускною здатністю 1 ГБіт/с задовольнятиме обслуговування поточного трафіку, забезпечуючи задовільну інтенсивність обслуговування та достатній резерв для обробки додаткових навантажень.

3.6 Вибір та обґрунтування застосування апаратних засобів

3.6.1 Опис автоматизованих робочих місць

Розробка автоматизованих робочих місць (АРМ) базується на аналізі мережевих завдань кожного з підрозділів підприємства. Особливу увагу приділено вибору обладнання та програмного забезпечення, що відповідають специфіці завдань кожного підрозділу. План розміщення мережевих елементів подано на рисунку 3.5.

Рисунок 3.5 – План розміщення мережевих компонентів

Таблиця 3.2 – Характеристики автоматизованих робочих місць

Відділ	Апаратне забезпечення			Програмне забезпечення	
Державне підприємство «Антонов»					
Адміністративний відділ	ПК	Процесор	Intel Core i5-13400	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 16 ГБ 3200	Офісні пакети	Microsoft 365
		ПЗУ	Samsung 970 EVO Plus NVMe 500 ГБ	Системи документообігу	M.E.Doc
	Периферія	Монітор	Dell P2422H	Засоби комунікації	Zoom, Microsoft Teams
		Принтер	HP LaserJet M141a	Системи управління проектами	Microsoft Project, Worksection
		Гарнітура	Sennheiser PC 3 Chat		
		Телефон	Cisco CP-7960G		
Маркетинговий відділ	ПК	Процесор	AMD Ryzen 5 5600X	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 32 ГБ 3200	Аналітичні платформи	Google Analytics, Tableau
		ПЗУ	Samsung 970 QVO SSD 2 ТБ	Відеоредактори	Adobe Premiere Pro, DaVinci Resolve
		Відеокарта	NVIDIA GeForce RTX 3060	Графічні редактори	Adobe Photoshop, Adobe Illustrator, CorelDRAW
	Периферія	Монітор	LG UltraFine 27UL850	Засоби управління рекламними кампаніями	Facebook Ads Manager, Google Ads.
		Плотер	HP DesignJet T230		
		Гарнітура	Sennheiser PC 3 Chat	CRM-системи	HubSpot, HelpCrunch
		Телефон	Cisco CP-7960G		
Юридичний відділ	ПК	Процесор	Intel Core i3-12100	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 16 ГБ 3200	Офісні пакети	Microsoft 365, Adobe Acrobat
		ПЗУ	Samsung 970 EVO Plus NVMe 500 ГБ	Системи документообігу	M.E.Doc

Продовження таблиці 3.2

Відділ	Апаратне забезпечення			Програмне забезпечення	
Юр. відділ	Периферія	Монітор	Dell P2422H	Засоби комунікації	Zoom, Microsoft Teams
		Принтер	HP LaserJet M141a	Корпоративне сховище	Nextcloud
		Гарнітура	Sennheiser PC 3 Chat		
		Телефон	Cisco CP-7960G		
Технічна підтримка (КС)	ПК	Процесор	Intel Xeon E-2236	ОС	Debian GNU/Linux 12.8
		ОЗП	32 ГБ ECC RAM	Системи моніторингу	Zabbix, Nagios, PRTG
		ПЗУ	WD 8 ТБ HDD, Samsung 970 EVO Plus NVMe 500 ГБ		
	Периферія	Монітор	Dell P2422H	Системи резервного копіювання	Veeam
		Принтер	HP LaserJet M141a		
		Гарнітура	Sennheiser PC 3 Chat	Засоби віддаленого доступу	TeamViewer, AnyDesk
		Телефон	Cisco CP-7960G		
Авіакомпанія «Авіалінії Антонова»					
Відділ з обслуговування клієнтів	ПК	Процесор	Intel Core i5-12400	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 16 ГБ 3200	Офісні пакети	Microsoft 365, Adobe Acrobat
		ПЗУ	Samsung 970 EVO Plus NVMe 500 ГБ		
	Периферія	Монітор	Dell P2422H	CRM (Управління відносинами з клієнтами)	KeyCRM, NetHunt
		Принтер	HP LaserJet M141a		
		Гарнітура	Sennheiser PC 3 Chat	Корпоративне сховище	Nextcloud
		Телефон	Cisco CP-7960G		

Продовження таблиці 3.2

Відділ	Апаратне забезпечення			Програмне забезпечення	
Логістичний відділ	ПК	Процесор	Intel Core i5-12600K	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 16 ГБ 3200		
		ПЗУ	Samsung 970 EVO Plus NVMe 500 ГБ	Система управління логістикою	TMS (CargoWise)
	Периферія	Монітор	Dell P2422H		
		Принтер	HP LaserJet M141a	Планування ресурсів підприємства	SAP ERP
		Гарнітура	Sennheiser PC 3 Chat		
		Телефон	Cisco CP-7960G	Офісні пакети	Microsoft 365
Технічний відділ	ПК	Процесор	AMD Ryzen 7 5700X3D	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 32 ГБ 3200	Система моніторингу авіації	SkyTrac
		ПЗУ	Samsung 970 QVO SSD 2 ТБ		
		Відеокарта	NVIDIA Quadro P2200	Управління технічним обслуговуванням авіації	CAMP Systems
	Периферія	Монітор	LG UltraFine 27UL850		
		Принтер	HP LaserJet M141a	Система автоматизованого проектування і креслення	AutoCAD
		Гарнітура	Sennheiser PC 3 Chat		
		Телефон	Cisco CP-7960G	Системи управління проектами	Microsoft Project, Worksection

Кінець таблиці 3.2

Відділ	Апаратне забезпечення				Програмне забезпечення
Віддалений підрозділ авіакомпанії «Авіалінії Антонова»					
Логістичний відділ	ПК	Процесор	Intel Core i5-12600K	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 16 ГБ 3200	Система управління логістикою	TMS (CargoWise)
		ПЗУ	Samsung 970 EVO Plus NVMe 500 ГБ	Планування ресурсів підприємства	SAP ERP
	Периферія	Монітор	Dell P2422H		
		Принтер	HP LaserJet M141a	Офісні пакети	Microsoft 365
		Гарнітура	Sennheiser PC 3 Chat		
		Телефон	Cisco CP-7960G		
Операційний відділ	ПК	Процесор	Intel Core i7-12700	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 16 ГБ 3200	Система автоматизованого проектування і креслення	AutoCAD
		ПЗУ	Samsung 970 EVO Plus NVMe 500 ГБ		
	Периферія	Монітор	Dell P2422H		
		Принтер	HP LaserJet M141a		
		Гарнітура	Sennheiser PC 3 Chat	Системи управління проектами	Microsoft Project, Worksection
		Телефон	Cisco CP-7960G		
Технічний відділ	ПК	Процесор	AMD Ryzen 7 5700X3D	ОС	Windows 11 Pro
		ОЗП	Kingston DDR4 32 ГБ 3200	Система моніторингу авіації Управління технічним обслуговуванням авіації Система автоматизованого проектування і креслення	SkyTrac CAMP Systems AutoCAD
		ПЗУ	Samsung 970 QVO SSD 2 ТБ		
	Периферія	Монітор	NVIDIA Quadro P2200		
		Принтер	LG UltraFine 27UL850	Системи управління проектами	Microsoft Project, Worksection
		Гарнітура	HP LaserJet M141a		
		Телефон	Sennheiser PC 3 Chat		

3.6.2 Опис інфраструктурного обладнання

Для побудови корпоративної комп'ютерної мережі підприємства було обрано комплекс мережевого обладнання з урахуванням сформульованих технічних вимог. Характеристики обладнання наведено в таблиці 3.3.

Таблиця 3.3 – Перелік мережевого обладнання

Тип пристрою	Модель	Характеристики	Кількість
Комутатори	Cisco Catalyst WS-C3560-24PS-E	24 портів Fast Ethernet, 4 SFP GigabitEthernet, підтримка PoE, VLAN, комутаційна здатність до 32 Гбіт/с	8
Маршрутизатори	Cisco Catalyst 8200-UCPE-1N8	6 портів Gigabit Ethernet	6
IP-камери	Dahua DH-IPC-HDW1230T1	2 Мп, інфрачервоне підсвічування до 30 м, підтримка PoE, кодек H.264, H.265	12
Сервер для відеоспостереження	HP ProLiant DL380 G10	2 процесори Intel Xeon Silver, 32 ГБ оперативної пам'яті, 12 од. WD Red Plus 6TB (RAID 6), 1 Гбіт NIC	1
Сервер IP-телефонії	HP ProLiant DL360 G9	2 процесори Intel Xeon, 32 ГБ оперативної пам'яті, 2 ТБ дискового простору	1
SFP-модуль	Cisco 1000BASE-T Copper SFP	1000BASE-T або 10/100/1000BASE-T	20
Мережева система керування доступом			
Турнікет-трипод зі зчитувачем RFID карт	ZKTeco TS1011 Pro	Пропускна здатність 30 чол/хв. Живлення AC110В / 220В, Wiegand	1
Біометричний термінал	ZKTeco SF100	Пам'ять шаблонів відбитків: 1500	1
Мережевий контролер	DS-K2708X	1x Ethernet 10/100/1000M, 100 В AC - 240 В AC, 200 Вт, Wiegand	3
Ригельний замок	ATIS Lock Mortise SS-R	–	28
ІЧ-датчик руху	Bosch ISC-BPQ2-W12	2 PIR-сенсори, площа дії 12x12 м.	35
Зчитувач RFID	ZKTeco KR503E	Тип карт: EM-Marine 125 кГц Інтерфейси зв'язку: Wiegand	28
Геркон	СМК-5Е	Врізний тип, черговий режим - 8мм, режим тривоги - 15мм.	28

3.7 Результати синтезу системи

У результаті синтезу системи було визначено ключові елементи її структури та принципи взаємодії між ними. Розрахунки комп'ютерної системи показали, що

пропускна здатність мережі повинна становити не менше 410,392 Мбіт/с. Мережа з пропускною здатністю 1 Гбіт/с забезпечить ефективне оброблення трафіку з мінімальними затримками (середня затримка кадру – 14,56 мкс). Для системи відеонагляду необхідний обсяг дискового масиву складає 51,89 ТБ, що включає 30% резерву та 15% для обслуговування дисків. Вибір накопичувачів ємністю 6 ТБ відповідає вимогам до надійності та надмірності. Результати розрахунків гарантують ефективне функціонування системи при пікових навантаженнях.

Обрані технічні засоби, такі як сервери, мережеві пристрої, відеокамери та засоби контролю доступу, відповідають вимогам і забезпечують надійну роботу системи. Концепція модульності дозволяє легко адаптувати систему до змінних потреб підприємства, забезпечуючи її довгострокову ефективність.

Впроваджені рішення спрямовані на створення безпечного та функціонального середовища, що відповідає вимогам підприємства до сучасної комп'ютерної інфраструктури.

4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Призначення й область застосування програмного забезпечення

Програмне забезпечення призначене для проведення тестування продуктивності мереж за допомогою протоколу RTP (Real-time Transport Protocol). Воно дозволяє вимірювати важливі параметри мережі, такі як затримка, джиттер, швидкість передачі даних та інші метрики, які критичні для оцінки якості з'єднання в реальному часі. Тестування включає вимірювання якості передачі голосових і відеоданих, що особливо актуально для систем, що використовують VoIP або відеоконференції.

Область застосування.

Програмне забезпечення використовується для загального моніторингу і тестування мереж, де потрібна оцінка кількості втрат пакетів, джиттера, швидкості передачі і затримок. Це важливо для інженерів та адміністраторів мереж, які працюють над покращенням мережевої інфраструктури та підтримкою високої якості зв'язку.

Програмне забезпечення може бути використано для тестування внутрішніх корпоративних мереж і серверних рішень, де важливо забезпечити безперервність зв'язку і ефективну передачу даних для внутрішніх комунікацій, таких як відеоконференції або внутрішні голосові дзвінки.

4.2 Постановка завдання на розробку програми

Метою розробки програмного забезпечення є створення інструменту для моніторингу та аналізу мережевої продуктивності, зокрема тестування якості з'єднань у реальному часі для систем, які використовують RTP (Real-time Transport Protocol). Програма повинна забезпечити можливість вимірювання ключових параметрів, таких як затримка, джиттер, втрата пакетів, пропускна здатність та інші, що впливають на якість зв'язку в мережах IP-телефонії та відеоконференцій.

Основні функції та вимоги:

– Тестування продуктивності мережі: Здійснення тестування шляхом відправлення RTP-пакетів по мережі та збору статистики про їх доставку. Вимірювання затримки, джиттера, втрат пакетів та інших критичних показників.

– Інтерфейс користувача: Програма повинна мати простий і зручний інтерфейс для налаштування тестів, запуску і моніторингу процесу. Можливість запису результатів тестів у вигляді графіків та статистичних таблиць. Налаштування параметрів тестування (тривалість тесту, розмір датаграм).

– Підтримка різних платформ: Програма повинна бути сумісною з основними операційними системами, такими як Windows, GNU/Linux, MacOS та Android щоб забезпечити універсальність у використанні.

4.3 Обґрунтування технічних характеристик програм

Програма має використовувати протокол RTP для тестування при вимірюванні метрик мережі, що відповідатиме вимогам реальних застосувань, таких як IP-телефонія та відеоконференції.

Для забезпечення користувачеві зручного досвіду, програма має використовувати графічні інтерфейси, що дозволяє швидко налаштувати тести, запускати їх та отримувати результати. Це також включає інтеграції з бібліотеками для побудови графіків і таблиць наприклад, matplotlib, що полегшує відображення результатів у зрозумілому вигляді.

Для досягнення кросплатформенності буде використано програмне середовище, яке підтримує розробку для кількох операційних систем одночасно наприклад, Python з бібліотеками для кросплатформенних додатків. Також буде розроблено веб-інтерфейс та android-застосунок. Це дозволяє досягти сумісності програми з основними операційними системами.

4.4 Розробка програми

Програмне забезпечення має чотири основні компоненти:

– run.py: Включає сервер та клієнт, що використовуються для проведення тестів на швидкість з використанням RTP пакетів.

– rtperf.py: Забезпечує простий графічний інтерфейс для запуску тестів та відображення результатів. Має можливість формувати статистику у CSV-файлі та будувати графіки метрик.

– webtest.py: Створює веб-інтерфейс для запуску тестів через браузер.

– Android-додаток: Мобільний застосунок для Android, що дозволяє користувачам запускати сервер та клієнт на мобільному пристрої для тестування швидкості з'єднання, а також отримувати результати безпосередньо на мобільному пристрої, що робить процес тестування мережі більш гнучким.

4.5 Опис розробленої програми

4.5.1 Загальні відомості

Розроблена програма є інструментом для тестування продуктивності RTP (Real-Time Protocol) через UDP. Вона складається з кількох компонентів, включаючи серверну та клієнтську частини, які взаємодіють для вимірювання параметрів мережі, таких як затримка, джиттер та швидкість передачі даних. Android-застосунок є інтерфейсною оболонкою, що взаємодіє з Python-сценаріями для виконання основних операцій, таких як запуск серверу, підключення клієнта, відправка і прийом даних.

4.5.2 Функціональне призначення

Програма призначена для вимірювання продуктивності мережі при передачі медіаданих. Вона дозволяє імітувати передачу потокових даних, включаючи відправку та отримання RTP-пакетів для тестування мережі, вимірюючи такі параметри, як затримка, джиттер та швидкість передачі даних. Отримані результати можуть зберігатися у файлі CSV та бути подані у вигляді графіків.

4.5.3 Зв'язок програми з іншими програмами

Flask сервер надає веб-інтерфейс для взаємодії з користувачем. Веб-браузер є клієнтським інтерфейсом для цієї програми, через який користувач взаємодіє з веб-додатком.

4.5.4 Використовувані технічні засоби

Програма використовує кілька бібліотек Python, таких як socket для роботи з мережевими з'єднаннями, struct для упаковки та розпакування RTP-пакетів, а також threading для запуску паралельних процесів. Для збереження результатів тестів використовується бібліотека pandas, а для побудови графіків – matplotlib. Окремий веб-інтерфейс, розроблений за допомогою Flask, дозволяє користувачам запускати тести та переглядати результати через браузер.

При розробці Android-застосунку використано:

– Python-скрипти через бібліотеку Chaquopy для виконання серверної та клієнтської частини тесту.

– Android SDK – для створення основного інтерфейсу застосунку та взаємодії з користувачем.

4.5.5 Виклик і завантаження

Програма запускається в одному з двох режимів: серверному або клієнтському. Сервер очікує на вхідні RTP-пакети, вимірює параметри продуктивності мережі та зберігає ці дані у CSV файл. Клієнт відправляє RTP-пакети на сервер, збираючи статистику щодо кількості відправлених пакетів та обсягу переданих даних.

Android-застосунок починає свою роботу за допомогою MainActivity, з визначеним у activity_main.xml інтерфейсом програми.

4.5.6 Вхідні дані

Вхідні дані для програми включають IP-адресу сервера, порт, тривалість тесту та розмір payload для RTP-пакетів. Сервер приймає RTP-пакети від клієнта, а також спеціальний пакет, який сигналізує про завершення передачі даних.

Android-застосунок передає введені користувачем параметри в Python-скрипт через бібліотеку Chaquopy, де виконується логіка тесту.

4.5.7 Вихідні дані

Програма генерує кілька видів вихідних даних:

– Статистика продуктивності: включає показники, такі як затримка, джиттер, швидкість передачі даних, втрата пакетів та інші метрики.

– CSV-файл: зберігається статистика про тести у вигляді CSV-файлу, який містить час, затримку, джиттер та швидкість передачі.

– Графіки: після завершення тестування користувач може переглянути графіки, що відображають зміну параметрів продуктивності з часом.

4.5.8 Опис логічної структури програми

Сервер створюється за допомогою сокета, що слухає на вказаному хості і порту. Сервер чекає отримання RTP пакетів від клієнта. Кожен прийнятий пакет обробляється функцією process_packet, яка витягує з нього необхідну інформацію.

Клієнт ініціює з’єднання з сервером і починає відправляти RTP пакети через UDP на сервер. Кожен пакет містить часову мітку, яка дозволяє серверу обчислювати затримку між відправкою і отриманням пакета.

Коли сервер отримує пакет, він обчислює:

– Затримку – час, що пройшов між відправкою пакета на клієнті і його отриманням на сервері.

– Джиттер – варіативність затримки між послідовними пакетами. Для цього на сервері зберігаються попередні значення затримки, і на кожен новий пакет розраховується відхилення від попереднього.

– Швидкість передачі – вимірюється через розмір пакета і час між відправленнями, що дозволяє визначити ефективність каналу.

Після обчислення метрик затримки, джиттера та швидкості передачі на сервері, ці значення передаються в консоль через функцію виведення.

Результати вимірювань можуть бути збережені в CSV файл для подальшого аналізу. Функція log_results періодично записує дані в форматі поданому на рисунку 4.1.

Програма завершується після визначеного часу, сервер і клієнт коректно закривають сокети, завершуючи своє з’єднання.

Time (s)	Latency (n	Jitter (ms)	Speed (Mbps)
2	0.013859	0.00273	808.1192
4.000008	0.013925	0.003006	804.3012
6.000009	0.013827	0.003	810.0294
8.000015	0.014008	0.003226	799.5273

Рисунок 4.1 – Формат зберігання файлу статистики

Для роботи з програмою необхідно налаштувати та запустити сервер. Користувач має можливість вказати порт для прослуховування.

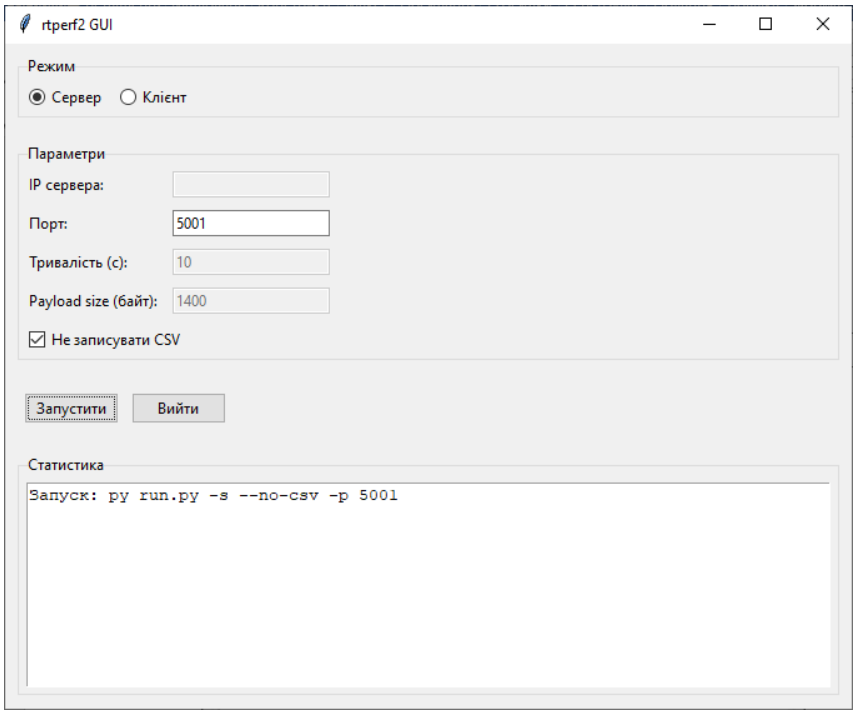


Рисунок 4.2 – Запущений сервер на пристрої з ОС Windows

Після того як сервер налаштовано, користувач переходить до налаштувань Клієнта. Тут задаються параметри для відправлення RTP пакетів на сервер, як IP-адреса сервера, порт, час тестування та розмір корисного навантаження пакету.

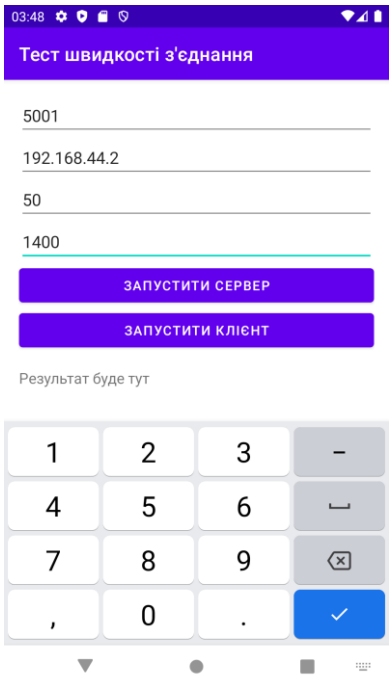


Рисунок 4.3 – Запуск тестування

Клієнт генерує пакети відповідно до заданих параметрів (частота, розмір, типи даних) і відправляє їх до сервера для обробки. Після завершення роботи клієнт передає звіт, а сервер результат тестування в інтерфейс користувача.

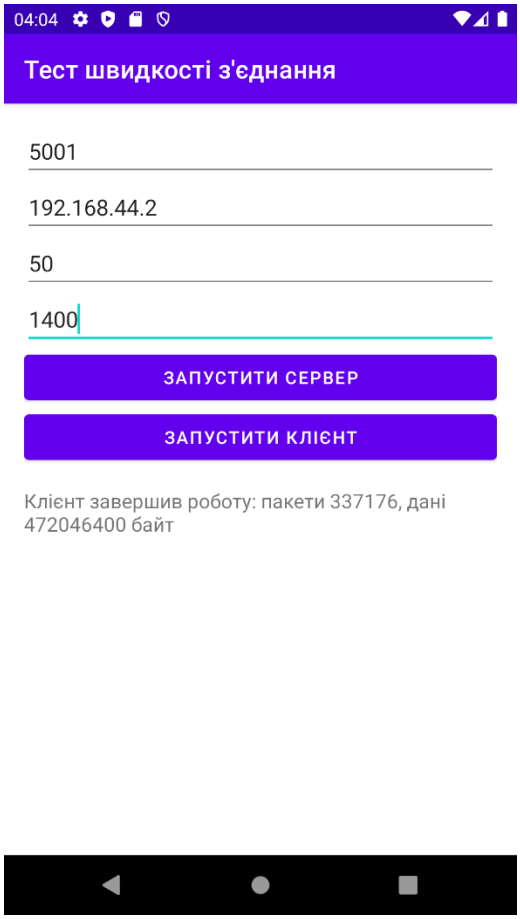


Рисунок 4.4 – Результат тестування клієнта

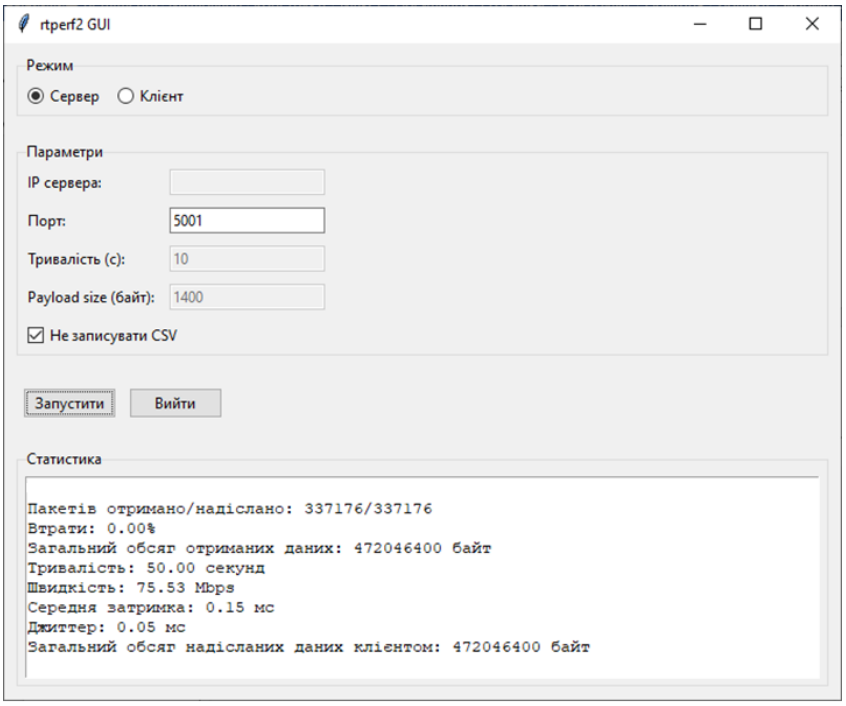


Рисунок 4.5 – Результат тестування сервера

Якщо в параметрах було вказано збереження проміжних результатів, сервер зберігає отримані дані в csv-файл та на основі отриманої статистики виводить графіки метрик.

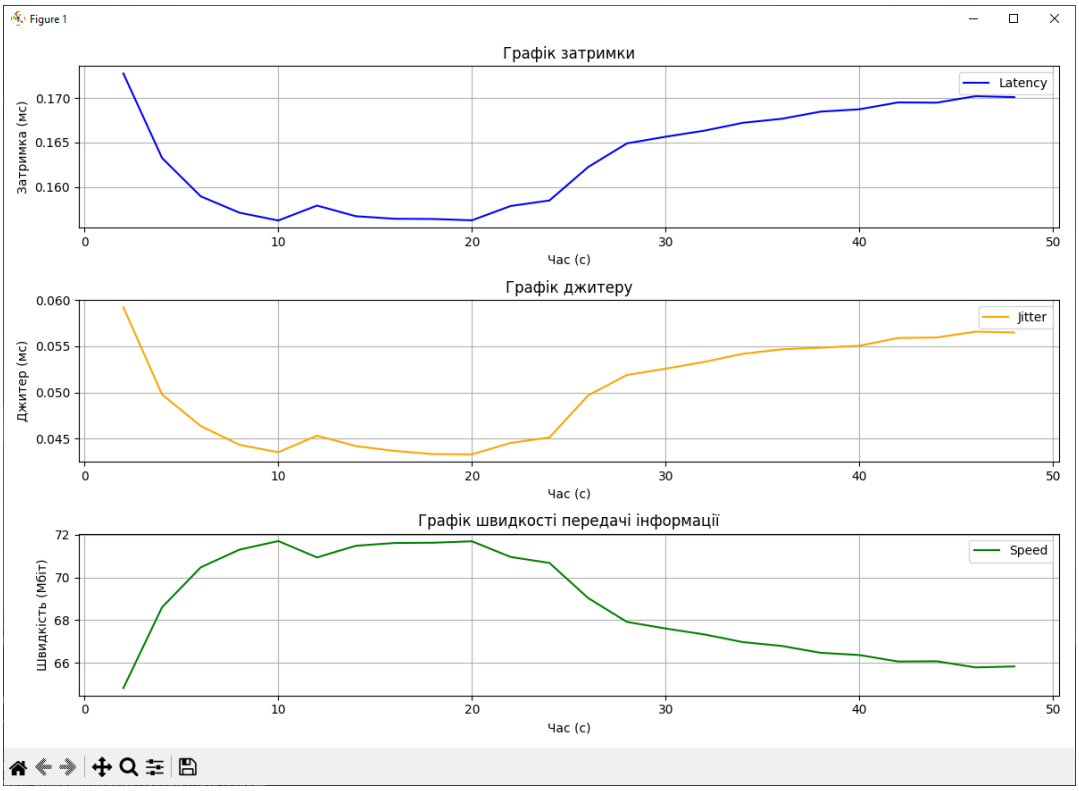


Рисунок 4.6 – Приклад графіку тестування

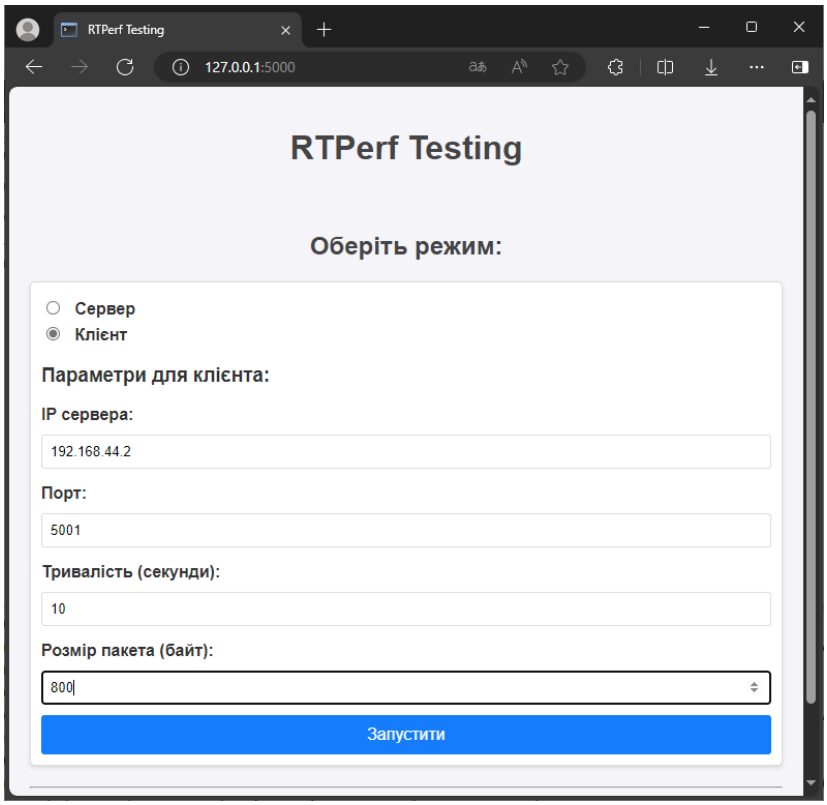


Рисунок 4.7 – Веб-інтерфейс програми

4.5.9 Розробка структурної схеми

Схема наведена на рисунку 4.9 чітко показує, як компоненти програми взаємодіють між собою через потоки даних, збереження результатів та відображення інформації користувачу.

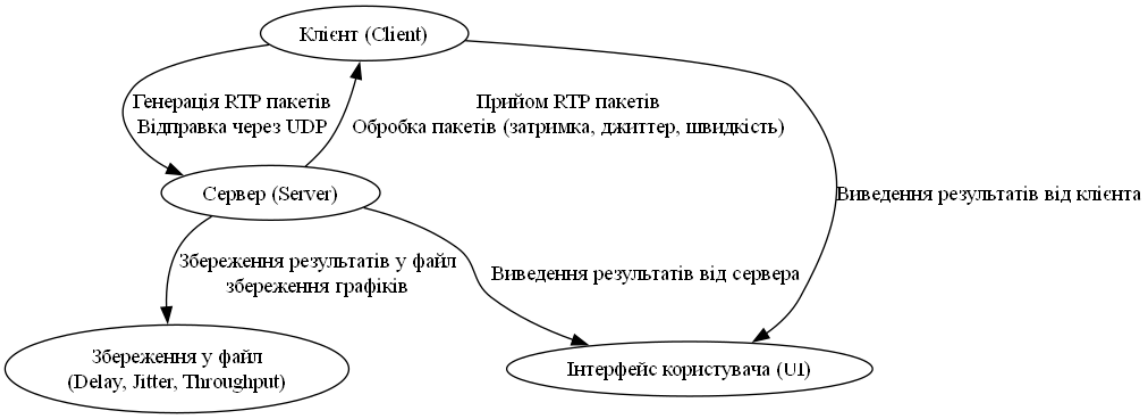


Рисунок 4.8 – Структурна схема програми

4.6 Очікувані техніко-економічні показники

Консольний інтерфейс: Час відгуку на запити користувача не повинен перевищувати 20 мс.

Графічний інтерфейс: Час відгуку на запити користувача не повинен перевищувати 50-100 мс.

Веб-інтерфейс: Час відгуку сервера на запити не повинен перевищувати 100-200 мс.

Android-застосунок: Час побудови інтерфейсу не більше 2 секунд. Час відгуку на запит користувача не більше 100-150 мс.

Ініціалізація тесту: Сервер має бути готовим до прийому підключень від клієнтів не більше ніж 200 мс. Час на ініціацію тесту – не більше 300 мс від запуску сервера.

Вірогідність помилок: Відсоток помилок у системі не повинен перевищувати 0.01% від загальної кількості тестів.

Стійкість до навантаження: Сервер повинен підтримувати до 10 одночасних клієнтських підключень без значних затримок.

Час побудови графіків: Графіки для відображення результатів тесту повинні генеруватися за не більше ніж 1-2 секунди.

5 ЕКСПЕРИМЕНТАЛЬНИЙ РОЗДІЛ

5.1 Проектування математичної моделі мережі як замкнутої системи масового обслуговування

На основі схеми поданої на рисунку 5.1 моделі комп'ютерної мережі була розроблена математична модель комп'ютерної мережі у вигляді замкнутої системи масового обслуговування.

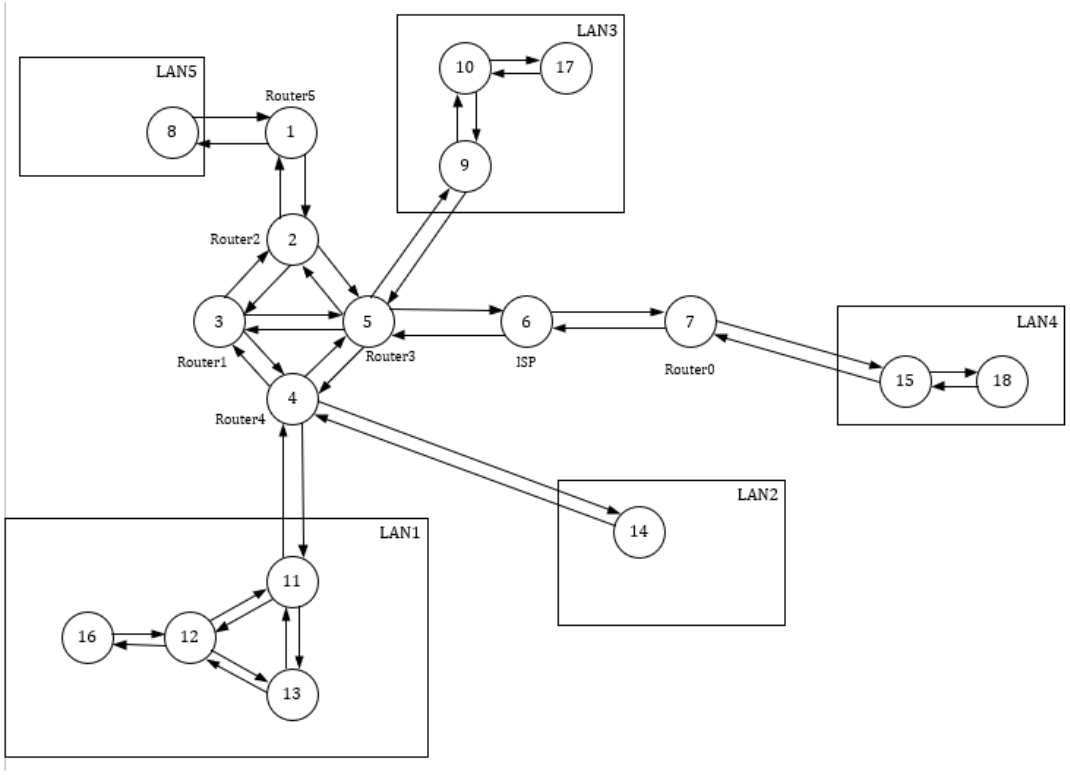


Рисунок 5.1 – Структура математичної моделі комп’ютерної мережі

У складі моделі комп'ютерної мережі вузли, що виконують функції комутаторів та маршрутизаторів, утворюють єдину структуру, де комутатори відповідають за обслуговування локальних мереж. Взаємодії між усіма компонентами цієї структури визначаються ймовірністю передачі пакета від одного вузла до іншого. Кожен вузол є системою масового обслуговування. Побудована матриця ймовірностей переходу пакету між вузлами системи відображає ймовірності того, що пакет переходить з одного вузла в інший, що

дозволяє моделювати рух даних у мережі та вивчати її поведінку. Ймовірність того, що вузол з'єднається сам із собою, дорівнює нулю.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	0	0.3	0	0	0	0	0	0.7	0	0	0	0	0	0	0	0	0	0
2	0.5	0	0.25	0	0.25	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0.33	0	0.33	0.33	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0.2	0	0.2	0	0	0	0	0	0.2	0	0	0.4	0	0	0	0
5	0	0.2	0.2	0.3	0	0.1	0	0	0.2	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0.5	0	0.5	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0.3	0	0	0	0	0	0	0	0	0.7	0	0	0
8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0.3	0	0	0	0	0.7	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0.3	0	0	0	0	0	0	0	0.7	0
11	0	0	0	0.3	0	0	0	0	0	0	0	0.35	0.35	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0.25	0	0.25	0	0	0.5	0	0
13	0	0	0	0	0	0	0	0	0	0	0.5	0.5	0	0	0	0	0	0
14	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0.4	0	0	0	0	0	0	0	0	0	0	0.6
16	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Рисунок 5.2 – Маршрутна матриця

Створено матрицю, що містить час обробки одного повідомлення на кожному з пристроїв мережі. Розраховано інтенсивність обслуговування для кожного з вузлів. Інтенсивність обслуговування визначається як обернене значення часу обробки повідомлення, що дає уявлення про швидкість обробки даних на кожному пристрої. Це дозволяє оцінити навантаження на вузли мережі та ефективність її роботи в цілому.

В результаті синтезу системи у частині 3 розраховано час обробки пакету, максимальна пропускна здатність мережі та очікуване максимальне навантаження на мережу.

Оцінка навантаження мережі буде виражена у відсотках, розраховуючи як відношення очікуваного максимального навантаження до максимальної пропускної здатності мережі:

$$\frac{410{,}392\ \text{Мбіт/с}}{1000\ \text{Мбіт/с}} = 41{,}04\% \approx 40\%$$

Таблиця 3.4 – Вхідні дані

Навантаженість мережі, N	Кількість вузлів, Nn	Час обробки пакету, T	Кількість конвеєрів у вузлі, m
40%	18 од.	14,56 мкс	1 од.
100%			

Прийнявши матриці PP2 та Q за систему лінійних алгебраїчних рівнянь методом LU-розкладу знайдено передаточні коефіцієнти.

$$e := \begin{pmatrix} 1 \\ 0.6 \\ 0.478 \\ 0.931 \\ 0.709 \\ 0.144 \\ 0.243 \\ 0.7 \\ 0.476 \\ 1.112 \\ 0.625 \\ 0.877 \\ 0.438 \\ 0.373 \\ 0.427 \\ 0.439 \\ 0.779 \\ 0.257 \end{pmatrix}$$

Рисунок 5.3 – Результат розрахунку передаточних коефіцієнтів

Сформовано матрицю, елементи якої відображають кількість конвеєрів для обробки пакетів у кожному вузлі системи масового обслуговування.

m =

	0
0	1
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1
13	1
14	1
15	1
16	1
17	1
18	1

Рисунок 5.4 – Кількість конвейєрів у кожному вузлі

Для розрахунків передбачається, що кожен пристрій має лише один конвеєр обробки пакетів. Матриця B представляє собою матрицю ймовірностей, яка показує, з якою ймовірністю в конкретному вузлі (рядку) може перебувати певна кількість пакетів у стані очікування обробки (номер стовпця).

Таблиця 3.5 – Ймовірності перебування пакетів у стані очікування обробки

Кількість пакетів, N Вузол, Nn	1	2	3	4
1	0.751	0.205	0.04	0.004151
2	0.85	0.134	0.015	0.0008966
3	0.881	0.109	0.009608	0.0004534
4	0.768	0.194	0.035	0.00335
5	0.823	0.155	0.021	0.001479
6	0.964	0.035	0.0009007	0.0000124
7	0.939	0.058	0.002541	0.00005956
8	0.826	0.153	0.02	0.001424
9	0.881	0.109	0.009529	0.0004477
10	0.723	0.223	0.049	0.005708
11	0.844	0.139	0.016	0.001013
12	0.844	0.139	0.016	0.001013

Кінець таблиці 5.2

Кількість пакетів, N / Вузол, Nn	1	2	3	4
13	0.891	0.101	0.008099	0.0003488
14	0.907	0.087	0.005911	0.0002154
15	0.894	0.098	0.007706	0.0003232
16	0.891	0.101	0.008135	0.0003512
17	0.806	0.167	0.025	0.001962
18	0.936	0.064	0.002908	0.00007046

Відповідно до алгоритму Бузена, для кожного вузла мережі розраховуються основні характеристики, зокрема:

– Кількість пакетів, що надходять до кожного вузла, визначає інтенсивність вхідного потоку пакетів у кожному з вузлів;

– Середня кількість пакетів, що перебувають у стані очікування обробки в кожному вузлі;

– Середній час обробки пакета у вузлі визначається як відношення середньої кількості пакетів, що очікують на обробку в кожному вузлі, до інтенсивності вхідного потоку пакетів у відповідному вузлі.

Таблиця 3.6 – Результати моделювання при навантаженні 40%

Вузол	Середня кількість пакетів які знаходяться у вузлі, одиниць	Інтенсивність потоку, що входить у вузли, пакетів/с	Середній час перебування пакета у вузлі, секунд
1	0.297	16340	0.0000182
2	0.166	9802	0.00001696
3	0.13	7809	0.0000166
4	0.274	15210	0.00001798
5	0.2	11580	0.00001729
6	0.037	2353	0.00001565
7	0.063	3970	0.00001592
8	0.197	11440	0.00001726
9	0.129	7777	0.00001659
10	0.337	18170	0.00001856
11	0.174	10210	0.00001704
12	0.174	14330	0.00001214
13	0.118	7156	0.00001648
14	0.099	6094	0.00001629

Кінець таблиці 5.3

Вузол	Середня кількість пакетів які знаходяться у вузлі, одиниць	Інтенсивність потоку, що входить у вузли, пакетів/с	Середній час перебування пакета у вузлі, секунд
15	0.115	6976	0.00001645
16	0.118	7172	0.00001648
17	0.223	12730	0.00001751
18	0.07	4199	0.00001669

Рисунок 5.5 – Діаграма інтенсивності вхідного потоку

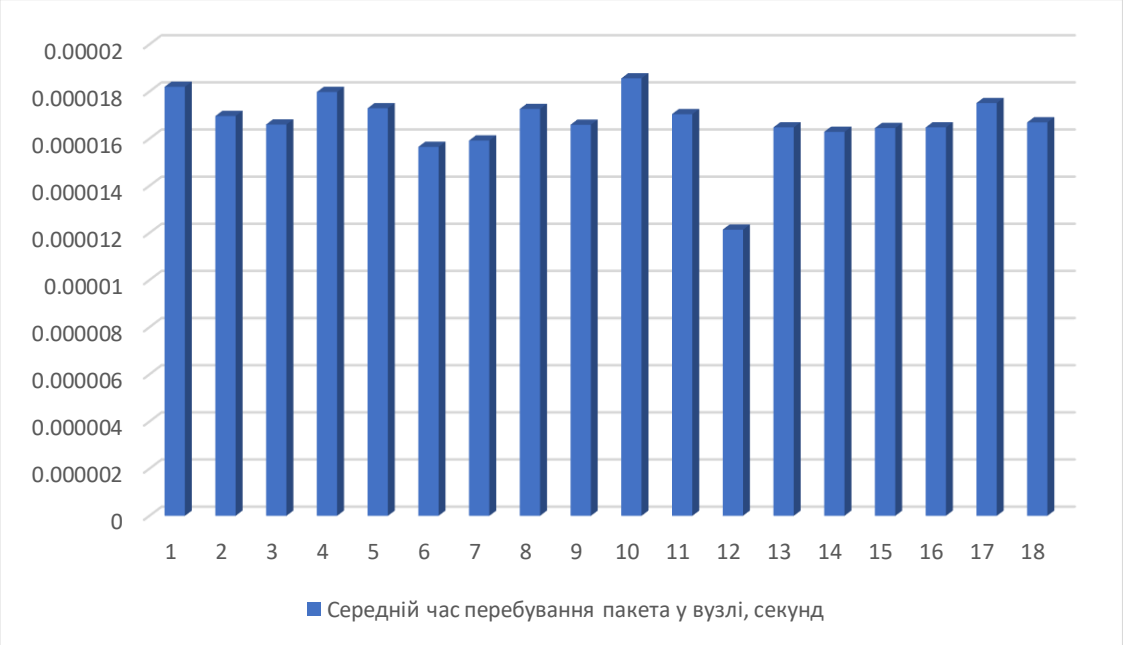


Рисунок 5.6 – Діаграма середнього часу обробки пакета

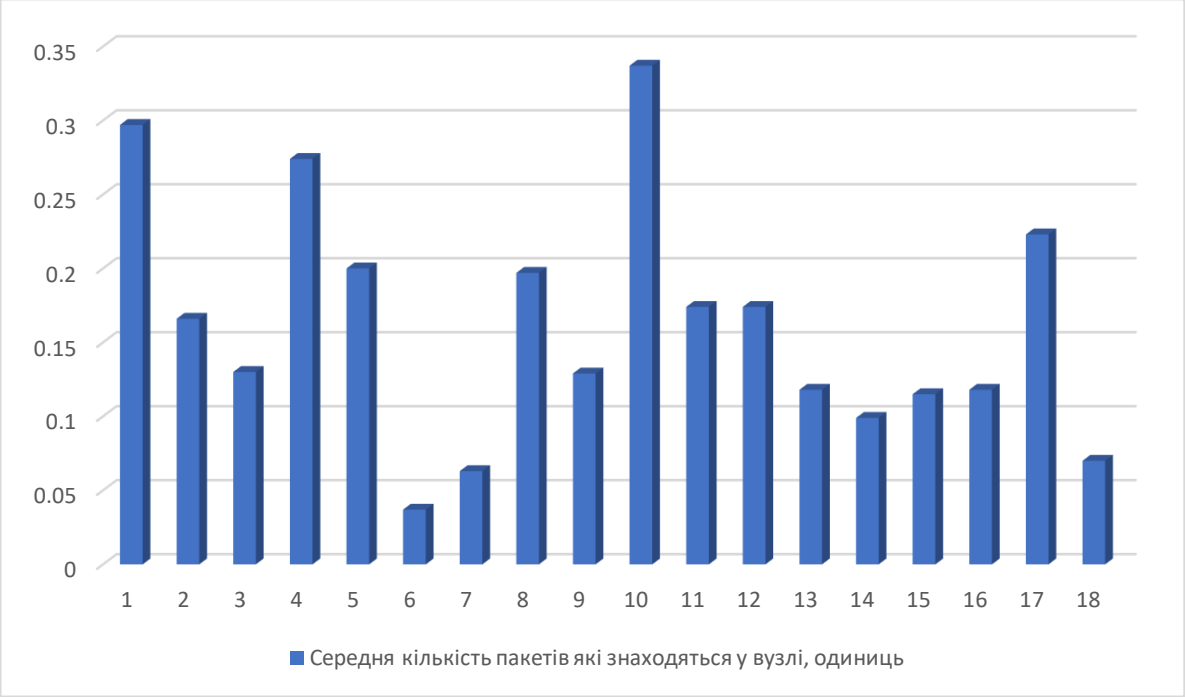


Рисунок 5.7 – Діаграма середньої кількості пакетів в очікуванні

Рисунок 5.9 показує ймовірність утворення черги в вузлах мережі при навантаженні 410 Мбіт/с. Аналізуючи ці ймовірності, можна зробити кілька висновків:

Для всіх вузлів ймовірність відсутності черги є найвищою. Ймовірності утворення черги значно менші, що свідчить про достатню продуктивність мережі. Вузли зазвичай не затримують багато пакетів одночасно, що вказує на їх ефективність у обробці. Загалом, діаграма демонструє рівномірний розподіл навантаження, а ймовірність скупчення пакетів залишається низькою.

В більшості вузлів ймовірність утворення черги з одного пакета досить висока, в той час як ймовірність утворення черги з більшої кількості пакетів значно зменшується. Це вказує на добре налаштовану мережу з низьким рівнем затримок та високою ефективністю обробки пакетів.

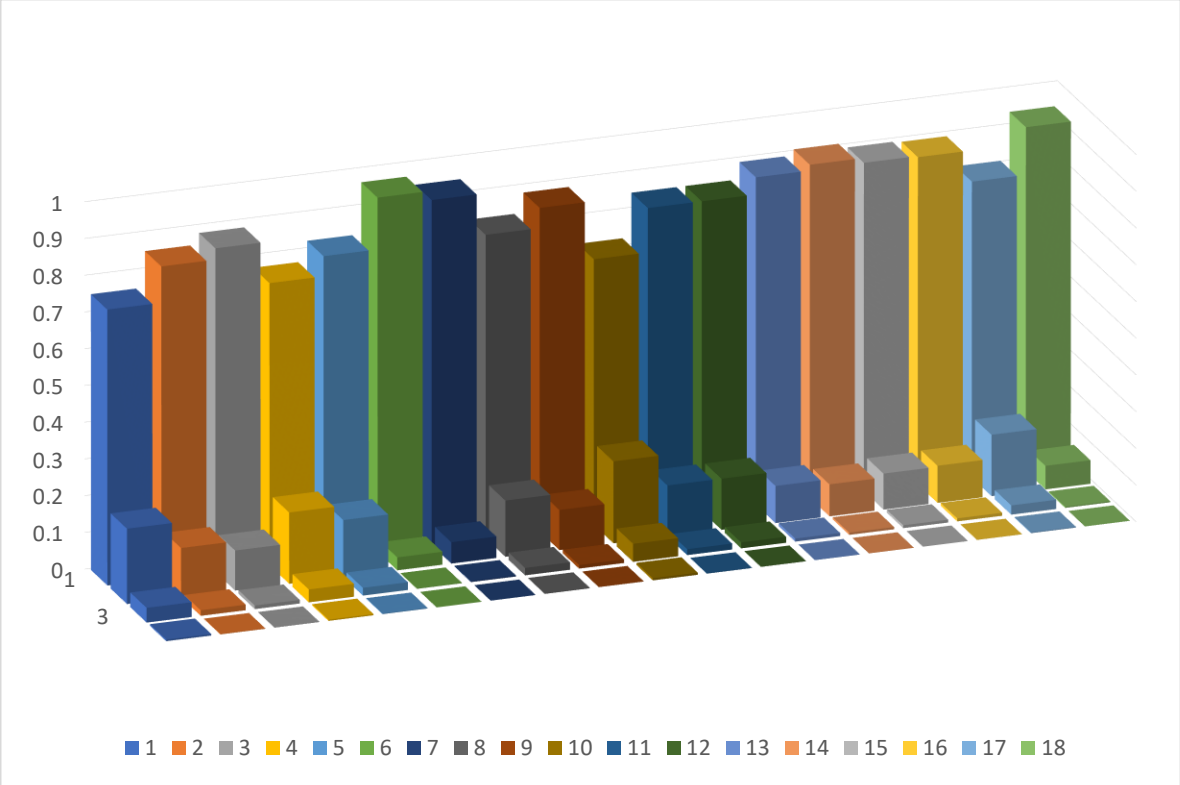


Рисунок 5.8 – Діаграма вірогідності утворення черги у вузлах мережі

Таблиця 3.7 – Результати моделювання при навантаженні 100%

Пристрій	Середня кількість пакетів які знаходяться у вузлі, одиниць	Інтенсивність потоку, що входить у вузли, пакетів/с	Середній час перебування пакета у вузлі, секунд
1	1.025	32390	0.00003166
2	0.46	19430	0.00002369
3	0.339	15480	0.0000219
4	0.905	30150	0.00003
5	0.586	22960	0.0000255
6	0.084	4664	0.00001804
7	0.15	7870	0.00001905
8	0.575	22670	0.00002535
9	0.337	15420	0.00002187
10	1.249	36020	0.00003467
11	0.488	20240	0.00002409
12	0.488	28400	0.00001716
13	0.303	14190	0.00002136
14	0.248	12080	0.00002054
15	0.293	13830	0.00002122
16	0.304	14220	0.00002138
17	0.676	25230	0.0000268
18	0.183	8324	0.00002199

Аналізуючи отримані дані, можна зробити висновок, що мережа досягає свого піку, при цьому зберігаючи задовільні результати навіть за високих навантажень. Хоча інтенсивність потоку та середня кількість пакетів у вузлах зростають, час перебування пакета залишається на задовільному рівні, що свідчить про доволі ефективну обробку пакетів.

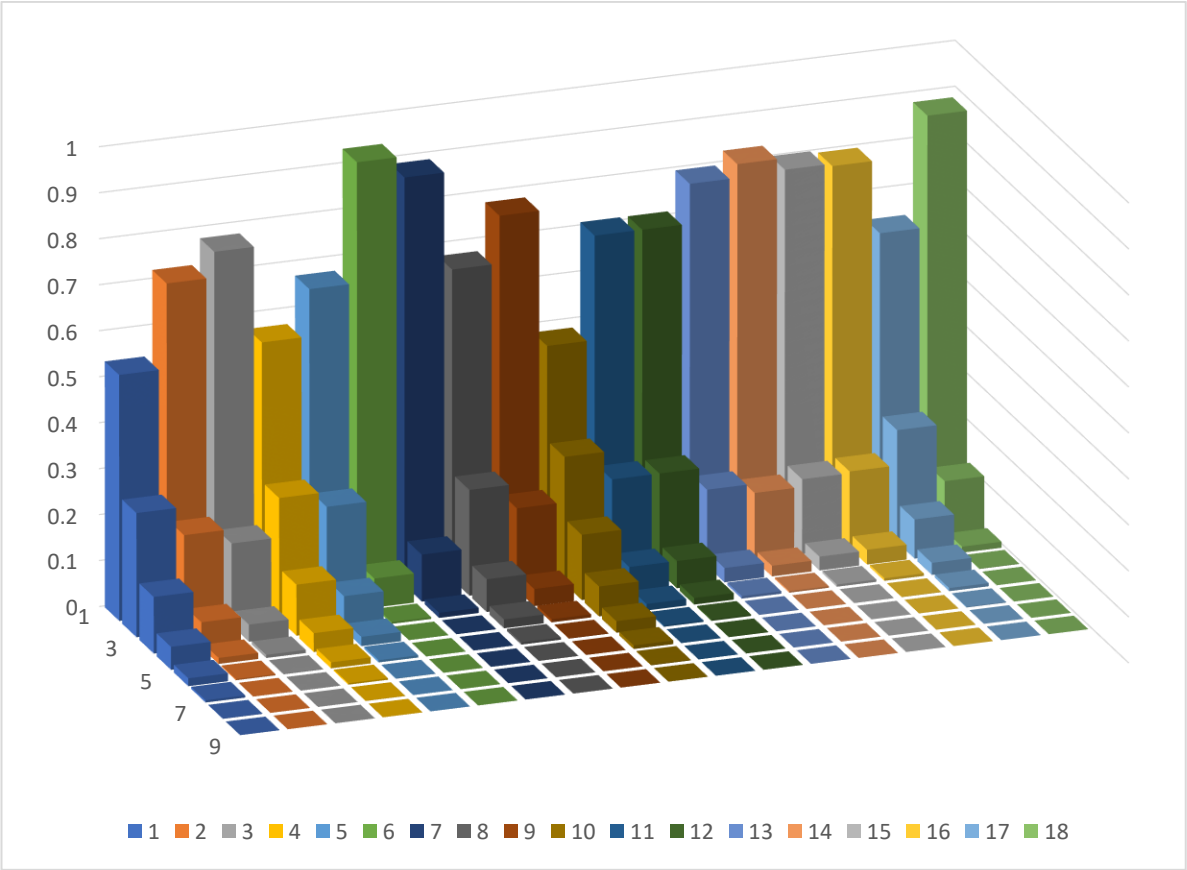


Рисунок 5.9 – Діаграма вірогідності утворення черги у вузлах мережі

Ймовірність утворення черги зростає, коли навантаження досягає 100%, що є очікуваним результатом. Порівняно з попереднім тестуванням, де ймовірність утворення черги була мінімальною, тепер спостерігається значне зростання ймовірностей утворення черги, що вказує на виникнення затримок у мережі. Це означає, що мережа вже не здатна підтримувати високий рівень обробки без виникнення затримок, і з часом ці затримки стають більш частими та тривалими.

Крім того, необхідно підкреслити нерівномірність розподілу ймовірностей між вузлами. В деяких вузлах ймовірність утворення черги значно вища, ніж в

інших, що свідчить про нерівномірне навантаження на мережеві пристрої. Це вказує на необхідність оптимізації навантаження в мережі для забезпечення більш рівномірного розподілу ресурсів і зниження ймовірності виникнення затримок у найбільш завантажених вузлах.

5.2 Тестування моделі мережі

Була створена модель мережі в середовищі EVE-NG з метою перевірки попередніх результатів. Вона включає образи операційних систем для хостів та мережевих пристроїв, що дозволяє проводити більш гнучкі тестування мереж з використанням спеціалізованого ПЗ. Створена модель дозволить на практиці застосувати розроблене програмне забезпечення та здійснити аналіз мережевого трафіку з використанням аналізатору трафіку Wireshark.

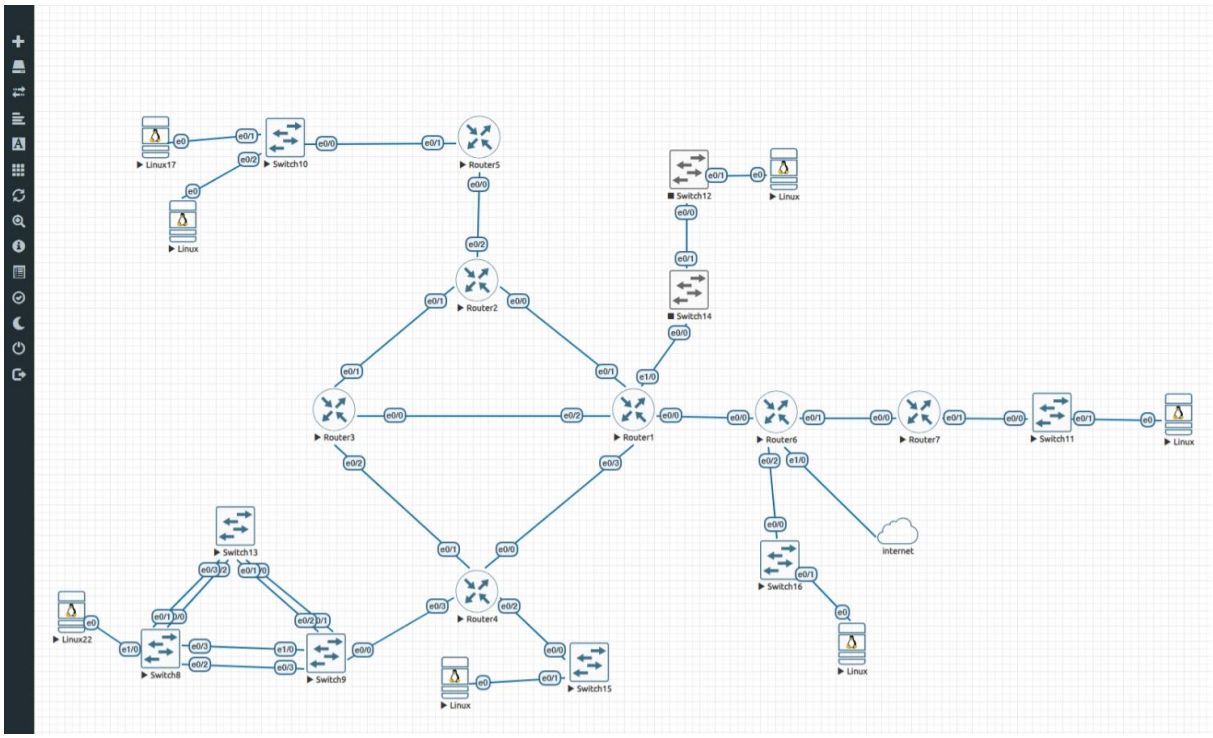


Рисунок 5.10 – Модель мережі

Для перехоплення пакетів між моделлю EVE-NG та сервером було використано технологію SPAN (Switched Port Analyzer), яка дозволяє копіювати трафік з порту комутатора для подальшого аналізу. У даному випадку

застосовувався комутатор Cisco Catalyst 3560, що забезпечив ефективне перенаправлення мережевого трафіку, а також нетбук Acer D270, який виконував роль пристрою для збору та аналізу перехоплених даних.

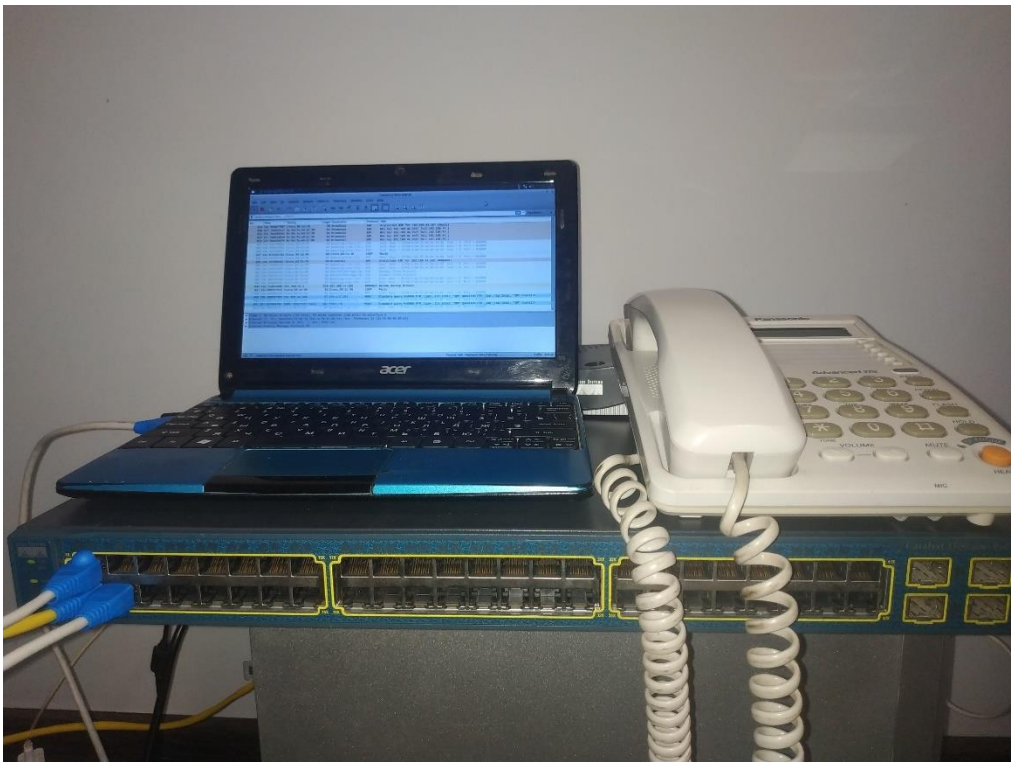


Рисунок 5.11 – Лабораторна конфігурація: Cisco Catalyst 3560 та Acer D270

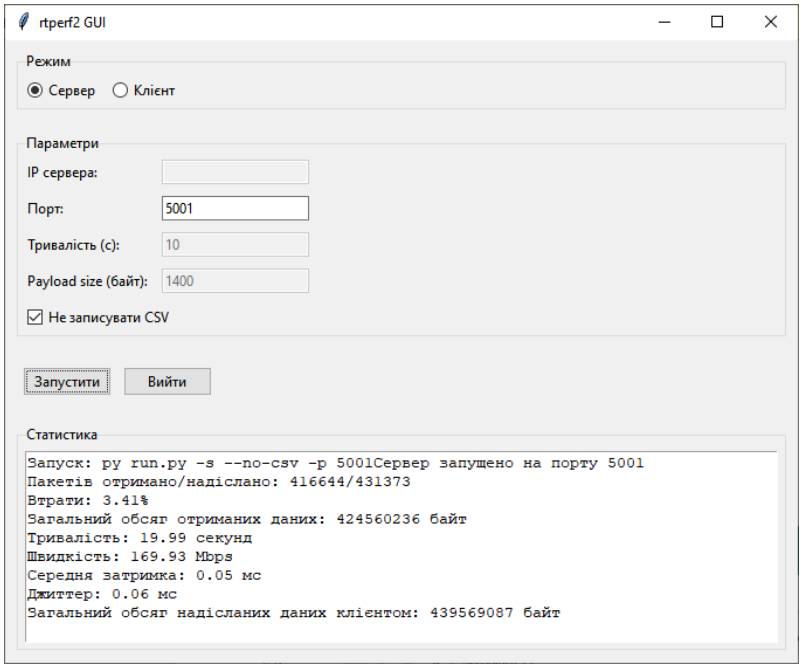


Рисунок 5.12 – Результат тестування мережі

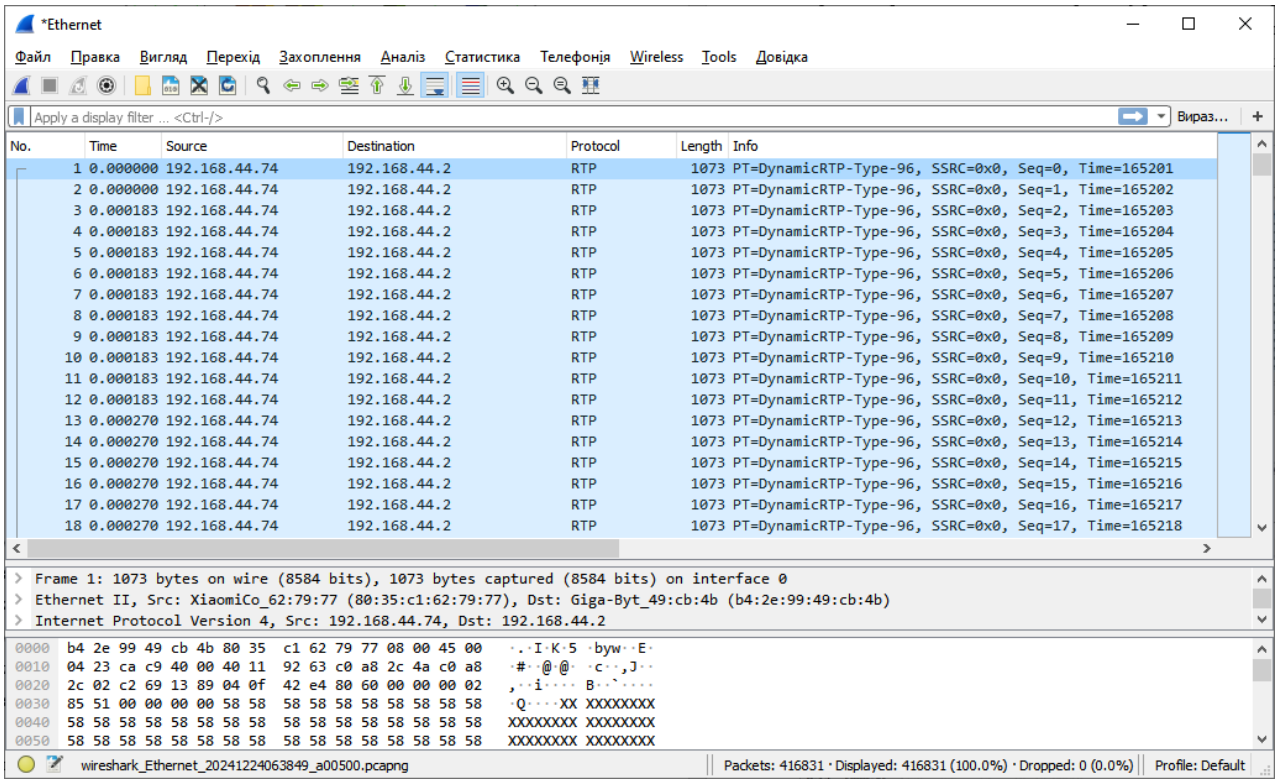


Рисунок 5.13 – Список перехоплених пакетів

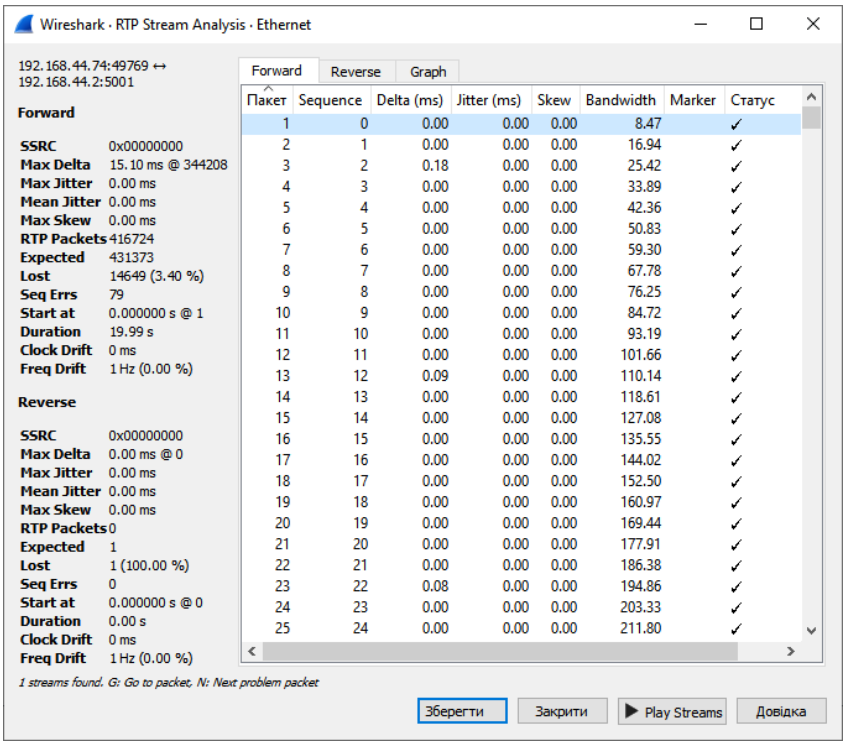
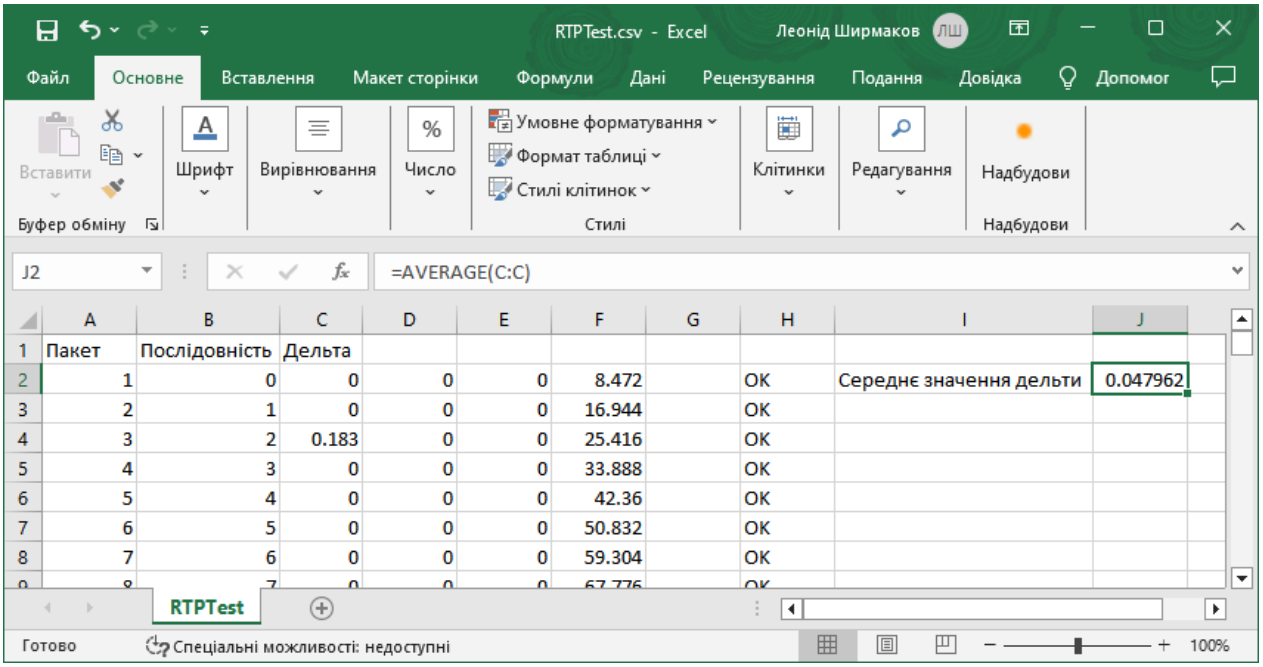


Рисунок 5.14 – Аналіз потоку RTP



	A	B	C	D	E	F	G	H	I	J
1	Пакет	Послідовність	Дельта							
2	1	0	0	0	0	8.472		OK	Середнє значення дельти	0.047962
3	2	1	0	0	0	16.944		OK		
4	3	2	0.183	0	0	25.416		OK		
5	4	3	0	0	0	33.888		OK		
6	5	4	0	0	0	42.36		OK		
7	6	5	0	0	0	50.832		OK		
8	7	6	0	0	0	59.304		OK		

Рисунок 5.15 – Розрахунок середнього значення дельти

5.3 Порівняння результатів дослідження

Протягом виконання роботи було отримано три значення затримок, що залежать від використаного методу: математичний розрахунок, математична модель та експеримент на моделі мережі. Кожне з цих значень відображає різні аспекти мережевих затримок: теоретичне значення базується на математичних розрахунках, модельоване – на результатах імітаційного моделювання, а практичне – при тестуванні моделі мережі.

Теоретична середня затримка пакету на вузлі при швидкості 410,392 Мбіт/с – 14,56 мкс.

Модельований час перебування пакету на вузлі при швидкості 410,392 Мбіт/с – 16.67 мкс.

Емпірична затримка при швидкості 169,93 Мбіт/с = 47.96 мкс.

Затримка передачі даних обернено пропорційна швидкості мережі. Це дає змогу масштабувати значення затримки між швидкостями.

$$T_2 = T_1 * \frac{S_1}{S_2},$$

де S_1 – швидкість передачі даних у мережі для мережі;

S_2 – швидкість передачі даних у між вузлами на моделі мережі;

T_1 – затримки передачі даних при швидкості S1;

T_2 – затримка передачі даних, що розраховується для швидкості S2.

$$T_2^{\text{теор.}} = 14{,}56 \text{ мкс} * \frac{410{,}392 \text{ Мбіт/с}}{169{,}93 \text{ Мбіт/с}} = 35{,}16 \text{ мкс}$$

$$T_2^{\text{модел.}} = 16{,}67 \text{ мкс} * \frac{410{,}392}{169{,}93} = 40{,}26 \text{ мкс}$$

$$\Delta_T^{\text{теор.}} = 1 - \frac{35{,}16 \text{ мкс}}{47{,}96 \text{ мкс}} \approx 0{,}2669 = 26{,}69\%$$

$$\Delta_T^{\text{модел.}} = 1 - \frac{40{,}26 \text{ мкс}}{47{,}96 \text{ мкс}} \approx 0{,}1605 = 16{,}05\%$$

ВИСНОВКИ

У результаті проведених досліджень було вирішено наукове завдання обґрунтування параметрів мережевих пристроїв комп’ютерної системи державного підприємства «Антонов» з врахуванням впровадження системи безпеки відеонагляду та IP телефонії. Вивчені параметри мережі, такі як пропускна здатність, затримки та ймовірність утворення черг в вузлах мережі, дозволили визначити оптимальні конфігурації для забезпечення стабільної роботи системи з урахуванням специфічних вимог безпеки та ефективної комунікації. Дослідження показали, що мережа досягла свого піку зберігаючи показники метрик в межах норми.

Результати дослідження показали, що практична затримка виявилася вищою за теоретичну та модельовану, що вказує на наявність факторів, не врахованих у теоретичних розрахунках та імітаційному моделюванні. Проте, метод статистичного імітаційного моделювання продемонстрував точніші результати порівняно з теоретичними розрахунками, оскільки модельована затримка виявилася лише на 16,05% меншою за практичну, що значно ближче до результатів мережі, ніж теоретичне значення, яке на 26,69% менше. Проявлена різниця у затримках може бути зумовлена кількома факторами, зокрема віртуалізацією мережевого обладнання, що спричиняє додаткові затримки через обмеження віртуальних інтерфейсів та обмежену пропускну здатність віртуальних мереж.

Практичне значення роботи полягає в розробці методичних рекомендацій для оптимізації налаштувань мережевих пристроїв при проектуванні системи безпеки відеонагляду та IP телефонії на підприємстві. Це дозволить забезпечити ефективну та безпечну роботу комп'ютерної мережі, відповідаючи вимогам щодо передачі великого обсягу відео- та голосового трафіку з мінімальними затримками. Крім того, на основі проведеного дослідження можна створити практичні рекомендації для вдосконалення мережевої інфраструктури підприємства в контексті впровадження новітніх технологій безпеки.

ПЕРЕЛІК ПОСИЛАНЬ

1. Aviation's Massive Impact On Global Economies. [Електронний ресурс] – Режим доступу: https://www.airlineratings.com/articles/aviations-massive-impact-on-global-economies-.

2. Аналіз стану світового ринку цивільної авіації та прогноз його розвитку в умовах нестабільного попиту на авіаперевезення. / Петрик, В. Л. Харків : Часопис економічних реформ, 2021.

3. A world loss event – EVER GIVEN. [Електронний ресурс] – Режим доступу: https://greco.services/a-world-loss-event-and-its-far-reaching-consequences-ever-given/.

4. Досвід застосування сільськогосподарської авіації. [Електронний ресурс] – Режим доступу: https://agrobusiness.com.ua/dosvid-zastosuvannia-silskohospodarskoi-aviatsii.

5. Закон України "Про розвиток літакобудівної промисловості". / Київ : Верховна Рада України, 2010.

6. Voice Service Interworking for PSTN and IP Networks. Лозанна : Institute for Computer Communications and Applicatins, 1999.

7. RTP Payload Format for G.711.0. / Ramalho, M. and Jones, P. s.l. : Huawei Technologies, 2015.

8. Maximizing Video Bandwidth: How to Ensure High-Quality Audio and Video Calls. [Електронний ресурс] – Режим доступу: https://trtc.io/blog/details/bandwidth.

9. Understanding VoIP Latency. [Електронний ресурс] – Режим доступу: https://www.occam.cx/audio-latency-for-voip-how-it-impacts-call-quality-ways-to-fix-it/.

10. What is low latency? And why is latency important?. [Електронний ресурс] – Режим доступу: https://www.pubnub.com/guides/whats-so-important-about-low-latency/.

11. The Heated Debate: Microsoft Teams vs. Zoom. [Електронний ресурс] – Режим доступу: https://shift.com/blog/apps-hub/the-heated-debate-microsoft-teams-vs-zoom/.

12. Understanding IP Surveillance. / Саннівейл : Fortinet, Inc, 2020.

13. Understanding SPAN, RSPAN and ERSPAN. [Електронний ресурс] – Режим доступу: https://community.cisco.com/t5/networking-knowledge-base/understanding-span-rspan-and-erspan/ta-p/3144951.

14. Analysis of Multiechelon Maintenance. / Madu, C. N. and Kuei, C.-H. Нью-Йорк : Elsevier Science Ltd, 1995.

15. SIP: Session Initiation Protocol. / Rosenberg, J., Schulzrinne, H. and Columbia, U. s.l. : Internet Engineering Task Force, 2002.

16. An Evaluation of Secure Real-time Transport Protocol (SRTP) Performance for VoIP. / Karne, R., Wijesinha, A. and Alexander, A. Тоусон : Towson University, 2009.

17. The Session Initiation Protocol (SIP) Digest Access Authentication Scheme. / Shekh-Yusef, R. s.l. : Internet Engineering Task Force, 2020.

18. RTP: A Transport Protocol for Real-Time Applications. / Schulzrinne, H., Casner, S. and Frederick, R. s.l. : Internet Engineering Task Force, 2003.

19. Bandwidth vs Speed. [Електронний ресурс] – Режим доступу: https://learningnetwork.cisco.com/s/blogs/a0D3i000002SKQrEAO/bandwidth-vs-speed.

20. What is Latency in Networking?. [Електронний ресурс] – Режим доступу: https://apposite-tech.com/latency/.

21. Voice over IP Security: Threat Analysis and Countermeasures. / Fysarakis, K. Лондон : Royal Holloway, University of London, 2007.

22. Analysis and Estimation of Playout Delay in VoIP Communications. / Sakuray, F., Hoto, R. and Mendes, L. s.l. : International Journal of Computer Science and Network Security, 2008.

23. What Is QoS?. [Електронний ресурс] – Режим доступу: https://www.spiceworks.com/tech/iot/articles/what-is-qos/.

24. Compare Traffic Policy and Traffic Shape to Limit Bandwidth. [Електронний ресурс] – Режим доступу: https://www.cisco.com/c/en/us/support/docs/quality-of-service-qos/qos-policing/19645-policevsshape.html.

25. What Is Quality Of Service (QoS) In Networking?. [Електронний ресурс] – Режим доступу: https://www.fortinet.com/resources/cyberglossary/qos-quality-of-service.

26. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. / Київ : Міністерство охорони здоров'я України, 1998.

27. QoS Requirements of Network Applications on the Internet. / Nong, Y. Темпе : Arizona State University, 2004.

28. G.711 Codec Technical Specifications. [Електронний ресурс] – Режим доступу: https://codecpro.com/en/codecs/codec_specifications/1/G711-annexes.

29. What is the maximum transmission unit (MTU)?. [Електронний ресурс] – Режим доступу: https://www.techtarget.com/searchnetworking/definition/maximum-transmission-unit.

30. defrag | Microsoft Learn. [Електронний ресурс] – Режим доступу: https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/defrag.

ДОДАТОК А
ТЕКСТ ПРОГРАМИ ТЕСТУВАННЯ ПРОДУКТИВНОСТІ МЕРЕЖІ

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ТЕСТУВАННЯ ПРОДУКТИВНОСТІ МЕРЕЖІ

Текст програми

804.02070743.24024-01 12 01

Листів 26

2024

АНОТАЦІЯ

Програма є інструментом для тестування продуктивності мережі через RTP (Real-Time Transport Protocol). Вона включає серверну та клієнтську частини, що дозволяють вимірювати метрики, такі як затримка, джиттер, втрати пакетів та швидкість передачі. Інтерфейс користувача на базі Tkinter дозволяє налаштовувати параметри тесту, а зібрані дані можна експортувати в CSV формат для подальшого аналізу. Програма забезпечує ефективне тестування мережевої продуктивності за різних умов підключення, що дозволяє оптимізувати передачу даних і покращити якість зв'язку.

Розроблено інтерфейс для платформи Android який реалізує взаємодію користувача з мережевими інструментами через простий графічний інтерфейс. Основна мета додатку – забезпечення можливості запуску мережевих тестів у режимі сервера або клієнта з використанням Python-скриптів. Програма поєднує функціонал Android SDK і Chaquopy – бібліотеки для інтеграції Python у мобільні додатки.

Додаток дозволяє користувачу вводити параметри тестування, такі як IP-адреса сервера, порт, тривалість тесту та розмір переданих даних (payload). На основі цих даних запускаються відповідні Python-скрипти для виконання тестів. У режимі сервера виконується запуск тестового сервера, а в режимі клієнта – надсилається запит до вказаного сервера для проведення мережевого аналізу.

ЗМІСТ

1 ПРОГРАМА ТЕСТУВАННЯ ПРОДУКТИВНОСТІ МЕРЕЖІ

1.1 Текст програми run.py

```
import socket
import struct
import time
import argparse
import csv
import threading
import queue
import pandas as pd

# Формат заголовка RTP пакету
RTP_HEADER_FORMAT = '!BBHII'
RTP_VERSION = 2  # Версія протоколу RTP
RTP_PAYLOAD_TYPE = 96   # Тип корисного навантаження (96 використовується для динамічних форматів)
DEFAULT_PAYLOAD_SIZE = 1400  # Розмір корисного навантаження у байтах (MTU 1500 мінус заголовки)

# Черга для зберігання метрик продуктивності
metrics_queue = queue.Queue()

# Створення RTP пакету
def create_rtp_packet(sequence_number, timestamp, payload):
# Формуємо заголовок RTP
rtp_header = struct.pack(RTP_HEADER_FORMAT,
 (RTP_VERSION << 6),  # Версія у старших 2 бітах
 RTP_PAYLOAD_TYPE,# Тип корисного навантаження
 sequence_number, # Номер послідовності
 timestamp,   # Часова мітка
 0)   # SSRC (ідентифікатор синхронізації)
return rtp_header + payload  # Повертаємо заголовок разом із корисним навантаженням

# Розбір RTP пакету
def parse_rtp_packet(packet):
rtp_header = packet[:12]  # Заголовок завжди займає 12 байтів
payload = packet[12:] # Корисне навантаження починається після заголовка
# Розпаковуємо заголовок
version, payload_type, sequence_number, timestamp, ssrc = struct.unpack(RTP_HEADER_FORMAT, rtp_header)
```

```
return version, payload_type, sequence_number, timestamp,
ssrc, payload

# Запис метрик продуктивності у CSV файл
def write_to_csv(metrics_queue):
data_list = []  # Локальний список для накопичення метрик
while True:
try:
data = metrics_queue.get(timeout=1)   # Отримання даних з
черги з тайм-аутом
if data == 'STOP':  # Сигнал для завершення запису
break
data_list.append(data)
if len(data_list) >= 100:  # Записуємо партії даних по 100
елементів
df = pd.DataFrame(data_list, columns=['Time (s)', 'Latency
(ms)', 'Jitter (ms)', 'Speed (Mbps)'])
df.to_csv('server_metrics.csv',   mode='a',   header=False,
index=False)
data_list.clear()  # Очищуємо список після запису
except queue.Empty:
continue  # Продовжуємо, якщо черга порожня

# Записуємо залишкові дані
if data_list:
df = pd.DataFrame(data_list, columns=['Time (s)', 'Latency
(ms)', 'Jitter (ms)', 'Speed (Mbps)'])
df.to_csv('server_metrics.csv',   mode='a',   header=False,
index=False)

# Запуск сервера RTP
def run_server(port, write_csv):
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)   #
Створення UDP сокета
sock.setsockopt(socket.SOL_SOCKET,  socket.SO_RCVBUF,  64  *
1024)  # Розмір буфера 64 КБ
sock.bind(('', port))  # Прив'язка сокета до порту
print("Сервер запущено на порту {}".format(port))

# Ініціалізація змінних для збирання статистики
packets_received = 0
total_data_received = 0
packet_times = []  # Час прибуття кожного пакету
client_packets_sent = 0
client_data_sent = 0
```

```
last_csv_time = None  # Час останнього запису у файл CSV

if write_csv:
# Ініціалізація CSV-файлу
with open('server_metrics.csv', mode='w', newline='') as
csv_file:
csv_writer = csv.writer(csv_file, delimiter=',')
csv_file.write("sep=,\n")  # Вказівка роздільника
csv_writer.writerow(['Time (s)', 'Latency (ms)', 'Jitter
(ms)', 'Speed (Mbps)'])

# Запуск потоку для запису у CSV
csv_thread = threading.Thread(target=write_to_csv,
args=(metrics_queue,), daemon=True)
csv_thread.start()

# Основний цикл отримання даних
try:
while True:
data, addr = sock.recvfrom(2048)  # Отримання даних від
клієнта
if data:
if data.startswith(b'END'):  # Сигнал завершення від клієнта
client_packets_sent, client_data_sent =
struct.unpack('!QQ', data[3:])
break

# Обробка першого пакета
if last_csv_time is None:
start_time = time.perf_counter()  # Час початку роботи
сервера
last_csv_time = start_time

# Розбір отриманого пакету
version, payload_type, sequence_number, timestamp, ssrc,
payload = parse_rtp_packet(data)
packets_received += 1
total_data_received += len(payload)
packet_times.append(time.perf_counter())  # Час прибуття
пакету

# Запис метрик кожні 2 секунди
current_time = time.perf_counter()
if write_csv and current_time - last_csv_time >= 2:
duration = current_time - start_time
```

```
bitrate = (total_data_received * 8) / duration / 1000000  # Розрахунок швидкості у Mbps

# Розрахунок затримки і джиттера
delays = [t - s for s, t in zip(packet_times, packet_times[1:])]
avg_delay = sum(delays) / len(delays) if delays else 0
jitter = sum(abs(d - avg_delay) for d in delays) / len(delays) if delays else 0

# Додавання метрик у чергу
metrics_queue.put([duration, avg_delay * 1000, jitter * 1000, bitrate])
last_csv_time = current_time
except KeyboardInterrupt:
print("Сервер зупинено вручну.")
finally:
if write_csv:
metrics_queue.put('STOP')  # Завершення потоку запису
csv_thread.join()

# Обчислення остаточної статистики
end_time = time.perf_counter()
duration = end_time - start_time
bitrate = (total_data_received * 8) / duration / 1000000
delays = [t - s for s, t in zip(packet_times, packet_times[1:])]
avg_delay = sum(delays) / len(delays) if delays else 0
jitter = sum(abs(d - avg_delay) for d in delays) / len(delays) if delays else 0
packet_loss = client_packets_sent - packets_received
packet_loss_percentage = (packet_loss / client_packets_sent) * 100 if client_packets_sent > 0 else 0

# Вивід статистики
print("Пакетів отримано/надіслано: {}/{}".format(packets_received, client_packets_sent))
print("Втрати: {:.2f}%".format(packet_loss_percentage))
print("Загальний обсяг отриманих даних: {} байт".format(total_data_received))
print("Тривалість: {:.2f} секунд".format(duration))
print("Швидкість: {:.2f} Mbps".format(bitrate))
print("Середня затримка: {:.2f} мс".format(avg_delay * 1000))
print("Джиттер: {:.2f} мс".format(jitter * 1000))
```

```
print("Загальний обсяг надісланих даних клієнтом: {} байт".format(client_data_sent))
```

1.2 Текст програми rtperf.py

```
import tkinter as tk
from tkinter import ttk, messagebox
import subprocess
import threading
import os
import time
import pandas as pd
import matplotlib.pyplot as plt

class rtperfGUI:
def __init__(self, root):
self.root = root
self.root.title("rtperf2 GUI")

# Основні змінні
self.mode = tk.StringVar(value="server")
self.server_ip = tk.StringVar()
self.port = tk.IntVar(value=5001)
self.test_duration = tk.IntVar(value=10)
self.payload_size = tk.IntVar(value=1400)
self.no_csv = tk.BooleanVar(value=False)

# Рамка режиму
mode_frame = ttk.LabelFrame(root, text="Режим")
mode_frame.grid(row=0, column=0, padx=10, pady=10, sticky="ew")

ttk.Radiobutton(mode_frame, text="Сервер", variable=self.mode, value="server", command=self.update_mode).grid(row=0, column=0, padx=5, pady=5)
ttk.Radiobutton(mode_frame, text="Клієнт", variable=self.mode, value="client", command=self.update_mode).grid(row=0, column=1, padx=5, pady=5)

# Рамка параметрів
param_frame = ttk.LabelFrame(root, text="Параметри")
param_frame.grid(row=1, column=0, padx=10, pady=10, sticky="nsew")
```

```
ttk.Label(param_frame, text="IP сервера:").grid(row=0, column=0, sticky="w", padx=5, pady=5)
self.server_ip_entry = ttk.Entry(param_frame, textvariable=self.server_ip, state="disabled")
self.server_ip_entry.grid(row=0, column=1, padx=5, pady=5)

ttk.Label(param_frame, text="Порт:").grid(row=1, column=0, sticky="w", padx=5, pady=5)
ttk.Entry(param_frame, textvariable=self.port).grid(row=1, column=1, padx=5, pady=5)

ttk.Label(param_frame, text="Тривалість (с):").grid(row=2, column=0, sticky="w", padx=5, pady=5)
self.duration_entry = ttk.Entry(param_frame, textvariable=self.test_duration, state="disabled")
self.duration_entry.grid(row=2, column=1, padx=5, pady=5)

ttk.Label(param_frame, text="Payload size (байт):").grid(row=3, column=0, sticky="w", padx=5, pady=5)
self.payload_entry = ttk.Entry(param_frame, textvariable=self.payload_size, state="disabled")
self.payload_entry.grid(row=3, column=1, padx=5, pady=5)

self.no_csv_check = ttk.Checkbutton(param_frame, text="Не записувати CSV", variable=self.no_csv, state="disabled")
self.no_csv_check.grid(row=4, column=0, columnspan=2, sticky="w", padx=5, pady=5)

# Кнопки
button_frame = ttk.Frame(root)
button_frame.grid(row=2, column=0, padx=10, pady=10, sticky="ew")

ttk.Button(button_frame, text="Запустити", command=self.run_rtperf).grid(row=0, column=0, padx=5, pady=5)
ttk.Button(button_frame, text="Вийти", command=root.quit).grid(row=0, column=1, padx=5, pady=5)

# Рамка для виводу статистики
stats_frame = ttk.LabelFrame(root, text="Статистика")
stats_frame.grid(row=3, column=0, padx=10, pady=10, sticky="nsew")
```

```
self.stats_text = tk.Text(stats_frame, wrap="word",
height=10, state="disabled")
self.stats_text.grid(row=0, column=0, padx=5, pady=5)

# Оновити доступність полів за замовчуванням
self.update_mode()

def update_mode(self):
"""Оновлює доступність параметрів залежно від вибраного
режиму."""
if self.mode.get() == "server":
self.server_ip_entry.config(state="disabled")
self.duration_entry.config(state="disabled")
self.payload_entry.config(state="disabled")
self.no_csv_check.config(state="normal")
else:
self.server_ip_entry.config(state="normal")
self.duration_entry.config(state="normal")
self.payload_entry.config(state="normal")
self.no_csv_check.config(state="disabled")

def run_rtperf(self):
"""Запускає rtperf2 через subprocess і відображає результати
в GUI."""
mode = self.mode.get()
args = ["py", "run.py"]

if mode == "server":
args.append("-s")
if self.no_csv.get():
args.append("--no-csv")
elif mode == "client":
if not self.server_ip.get():
messagebox.showerror("Помилка", "Будь ласка, вкажіть IP-
адресу сервера.")
return
args.extend(["-c", self.server_ip.get()])
args.extend(["-t", str(self.test_duration.get())])
args.extend(["--payload", str(self.payload_size.get())])

args.extend(["-p", str(self.port.get())])

self.stats_text.config(state="normal")
self.stats_text.delete("1.0", tk.END)
```

```
self.stats_text.insert(tk.END, "Запуск: {}".format("
".join(args)))
self.stats_text.config(state="disabled")

def run_process():
process = subprocess.Popen(args, stdout=subprocess.PIPE,
stderr=subprocess.PIPE, text=True)

if self.mode.get() == "client":
self.show_progress_window(self.test_duration.get())

for line in process.stdout:
self.stats_text.config(state="normal")
self.stats_text.insert(tk.END, line)
self.stats_text.see(tk.END)
self.stats_text.config(state="disabled")

process.wait()

if process.returncode != 0:
error_message = process.stderr.read()
messagebox.showerror("Помилка", f"Помилка виконання:
{error_message}")
else:
if self.mode.get() == "server" and not self.no_csv.get():
self.show_graphs()


threading.Thread(target=run_process, daemon=True).start()

def show_progress_window(self, duration):
"""Створює окреме вікно для відображення прогресу
клієнтського тесту."""
progress_window = tk.Toplevel(self.root)
progress_window.title("Прогрес виконання")

progress_label = ttk.Label(progress_window, text="Прогрес
виконання тесту")
progress_label.pack(pady=10)

progress_bar = ttk.Progressbar(progress_window,
orient="horizontal", mode="determinate", maximum=100)
progress_bar.pack(fill="x", padx=10, pady=10)

def update_progress():
```

```
start_time = time.perf_counter()
while True:
elapsed_time = time.perf_counter() - start_time
progress = min((elapsed_time / duration) * 100, 100)
progress_bar.config(value=progress)
if progress >= 100:
break
time.sleep(0.1)
progress_window.destroy()

threading.Thread(target=update_progress,
daemon=True).start()

def show_graphs(self):
"""Читає дані з CSV і будує графіки."""
try:
data = pd.read_csv("server_metrics.csv", skiprows=1)    #
Пропустити "sep=,"
time_data = data["Time (s)"]
latency_data = data["Latency (ms)"]
jitter_data = data["Jitter (ms)"]
speed_data = data["Speed (Mbps)"]

# Функція для згладжування даних
def smooth_data(x, y):
x_new = np.linspace(x.min(), x.max(), 300)  # Більше точок
для гладкої лінії
spline = make_interp_spline(x, y)
y_new = spline(x_new)
return x_new, y_new

plt.figure(figsize=(12, 8))

# Latency
plt.subplot(3, 1, 1)
plt.plot(time_data,     latency_data,     label="Latency",
color="blue")
plt.xlabel("Час (с)")
plt.ylabel("Затримка (мс)")
plt.title("Графік затримки")
plt.grid(True)
plt.legend()

# Jitter
plt.subplot(3, 1, 2)
```

```
plt.plot(time_data,      jitter_data,      label="Jitter",
color="orange")
plt.xlabel("Час (с)")
plt.ylabel("Джитер (мс)")
plt.title("Графік джитеру")
plt.grid(True)
plt.legend()

# Speed
plt.subplot(3, 1, 3)
plt.plot(time_data,       speed_data,       label="Speed",
color="green")
plt.xlabel("Час (с)")
plt.ylabel("Швидкість (Мбіт)")
plt.title("Графік швидкості передачі інформації")
plt.grid(True)
plt.legend()

plt.tight_layout()
plt.show()

except Exception as e:
messagebox.showerror("Помилка",  f"Не  вдалося  побудувати
графіки: {e}")

if __name__ == "__main__":
root = tk.Tk()
app = rtperfGUI(root)
root.mainloop()
```

1.3 Текст програми webtest.py

```
# Імпортуємо необхідні модулі
from flask import Flask, render_template, request, redirect,
url_for, flash
import subprocess
import threading
import pandas as pd
import matplotlib.pyplot as plt
import os

# Ініціалізація Flask-додатка
app = Flask(__name__)
app.secret_key = "secret_key"  # Ключ для Flash-повідомлень
```

```
# Додаткові параметри
PORT = 5001  # Порт за замовчуванням
CSV_FILE = "server_metrics.csv"  # Назва CSV-файлу зі статистикою

@app.route("/")
def index():
"""Головна сторінка."""
return render_template("index.html")

@app.route("/run", methods=["POST"])
def run_rtperf():
"""Обробляє запуск інструменту rtperf у серверному або клієнтському режимі."""
mode = request.form["mode"]  # Режим роботи: сервер або клієнт
server_ip = request.form.get("server_ip")  # IP-адреса сервера
port = request.form.get("port", PORT)  # Порт (за замовчуванням 5001)
duration = request.form.get("duration", 10)  # Тривалість тесту
payload = request.form.get("payload", 1400)  # Розмір payload
no_csv = "no_csv" in request.form  # Чи вимикати запис у CSV
args = ["py", "run.py"]  # Команда для запуску скрипту

# Формування аргументів для запуску
if mode == "server":
args.append("-s")  # Запуск у серверному режимі
if no_csv:
args.append("--no-csv")  # Відключення CSV
elif mode == "client":
if not server_ip:
flash("IP сервера обов'язковий для клієнта.", "error")
return redirect(url_for("index"))
args.extend(["-c", server_ip, "-t", str(duration), "--payload", str(payload)])

args.extend(["-p", str(port)])

# Запуск у фоновому потоці
thread = threading.Thread(target=run_subprocess, args=(args,))
thread.start()
```

```
flash(f"Режим {mode} запущено. Очікуйте завершення...",
"success")
return redirect(url_for("index"))

def run_subprocess(args):
"""Виконує команду у фоновому режимі та зберігає лог-
файли."""
try:
process = subprocess.Popen(
args,
stdout=subprocess.PIPE,
stderr=subprocess.PIPE,
text=True
)
with open("output.log", "w") as log_file:
for line in process.stdout:
log_file.write(line)

error = process.stderr.read()
if error:
with open("error.log", "w") as error_file:
error_file.write(error)

process.wait()
except Exception as e:
print(f"Помилка запуску subprocess: {e}")

@app.route("/logs")
def logs():
"""Показує вміст лог-файлу."""
if os.path.exists("output.log"):
with open("output.log", "r") as log_file:
content = log_file.read()
return render_template("logs.html", logs=content)
else:
return "Лог-файл відсутній."

@app.route("/graphs")
def graphs():
"""Генерує графіки на основі даних з CSV-файлу."""
if not os.path.exists(CSV_FILE):
flash("Файл статистики не знайдено.", "error")
return redirect(url_for("index"))
```

```
try:
# Завантаження даних із CSV
data = pd.read_csv(CSV_FILE, skiprows=1)
time_data = data["Time (s)"]
latency_data = data["Latency (ms)"]
jitter_data = data["Jitter (ms)"]
speed_data = data["Speed (Mbps)"]

# Побудова графіків
plt.figure(figsize=(12, 8))

# Графік затримки
plt.subplot(3, 1, 1)
plt.plot(time_data,      latency_data,      label="Latency",
color="blue")
plt.xlabel("Час (с)")
plt.ylabel("Затримка (мс)")
plt.title("Графік затримки")
plt.grid(True)
plt.legend()

# Графік джитеру
plt.subplot(3, 1, 2)
plt.plot(time_data,       jitter_data,       label="Jitter",
color="orange")
plt.xlabel("Час (с)")
plt.ylabel("Джитер (мс)")
plt.title("Графік джитеру")
plt.grid(True)
plt.legend()

# Графік швидкості
plt.subplot(3, 1, 3)
plt.plot(time_data,        speed_data,        label="Speed",
color="green")
plt.xlabel("Час (с)")
plt.ylabel("Швидкість (Мбіт)")
plt.title("Графік швидкості передачі інформації")
plt.grid(True)
plt.legend()

# Збереження графіка у файл
plt.tight_layout()
plt.savefig("static/graphs.png")
plt.close()
```

```
return                          render_template("graphs.html",
graph_url="/static/graphs.png")
except Exception as e:
flash(f"Помилка створення графіків: {e}", "error")
return redirect(url_for("index"))

# Запуск Flask-додатка
if __name__ == "__main__":
app.run(debug=True)
```

1.4 Текст програми templates/graphs.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta    name="viewport"    content="width=device-width,
initial-scale=1.0">
    <title>Графіки</title>
    <link                                    rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/css/bootstrap.min.css">
</head>
<body>
    <div class="container mt-5">
        <h1 class="mb-4">Графіки результатів</h1>

        <div class="text-center">
            <img   src="{{   graph_url   }}"   alt="Графік"
class="img-fluid border">
        </div>

        <a href="/" class="btn btn-primary mt-3">Повернутись
на головну</a>
    </div>

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

1.5 Текст програми templates/index.html

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<title>RTPerf Testing</title>
<style>
body {
font-family: Arial, sans-serif;
margin: 0;
padding: 20px;
background-color: #f4f4f9;
color: #333;
}
h1, h2 {
text-align: center;
color: #444;
}
form {
margin: 20px 0;
padding: 10px;
background: #fff;
border: 1px solid #ddd;
border-radius: 5px;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}
label {
display: block;
margin: 5px 0;
font-weight: bold;
}
input[type="text"], input[type="number"] {
width: 100%;
padding: 8px;
margin: 5px 0 10px;
border: 1px solid #ddd;
border-radius: 3px;
box-sizing: border-box;
}
input[type="radio"] {
margin-right: 10px;
}
button {
display: block;
width: 100%;
```

```
padding: 10px;
background-color: #007BFF;
color: #fff;
border: none;
border-radius: 3px;
cursor: pointer;
font-size: 16px;
}
button:hover {
background-color: #0056b3;
}
.link-button {
display: inline-block;
padding: 10px 15px;
margin: 10px 5px;
background-color: #28a745;
color: white;
text-decoration: none;
font-size: 16px;
border-radius: 5px;
text-align: center;
}
.link-button:hover {
background-color: #218838;
}
.messages {
margin: 20px 0;
padding: 10px;
border-radius: 5px;
}
.messages ul {
list-style-type: none;
padding: 0;
}
.messages li {
padding: 5px 0;
}
.messages .success {
background-color: #d4edda;
color: #155724;
}
.messages .error {
background-color: #f8d7da;
color: #721c24;
}
```

```
a {
color: #007BFF;
text-decoration: none;
}
a:hover {
text-decoration: underline;
}
.hidden {
display: none;
}
</style>
</head>
<body>
<h1>RTPerf Testing</h1>

<!-- Відображення повідомлень -->
<div class="messages">
{%               with               messages                    =
get_flashed_messages(with_categories=true) %}
{% if messages %}
<ul>
{% for category, message in messages %}
<li class="{{ category }}">{{ message }}</li>
{% endfor %}
</ul>
{% endif %}
{% endwith %}
</div>

<h2>Оберіть режим:</h2>
<form      id="modeForm"      method="POST"      action="{{
url_for('run_rtperf') }}">
<label>
<input     type="radio"     name="mode"     value="server"
onclick="toggleForm('server')" required> Сервер
</label>
<label>
<input     type="radio"     name="mode"     value="client"
onclick="toggleForm('client')"> Клієнт
</label>

<div id="serverParams" class="hidden">
<h3>Параметри для сервера:</h3>
<label for="port_server">Порт:</label>
```

```
<input type="number" id="port_server" name="port"
value="5001">
</div>

<div id="clientParams" class="hidden">
<h3>Параметри для клієнта:</h3>
<label for="server_ip">IP сервера:</label>
<input type="text" id="server_ip" name="server_ip">
<label for="port_client">Порт:</label>
<input type="number" id="port_client" name="port"
value="5001">
<label for="duration">Тривалість (секунди):</label>
<input type="number" id="duration" name="duration"
value="10">
<label for="payload">Розмір пакета (байт):</label>
<input type="number" id="payload" name="payload"
value="1400">
</div>

<button type="submit">Запустити</button>
</form>

<hr>
<a href="{{ url_for('logs') }}" class="link-
button">Переглянути логи</a>
<a href="{{ url_for('graphs') }}" class="link-
button">Переглянути графік</a>

<script>
function toggleForm(mode) {
document.getElementById('serverParams').classList.toggle('h
idden', mode !== 'server');
document.getElementById('clientParams').classList.toggle('h
idden', mode !== 'client');
}
</script>
</body>
</html>
```

2 ПРОГРАМА ТЕСТУВАННЯ ПРОДУКТИВНОСТІ МЕРЕЖІ ДЛЯ ПРИСТРОЇВ ОС ANDROID

2.1 Текст програми MainActivity.java

```
package ua.leonidshyrmakov.monitoring;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;
import com.chaquo.python.android.AndroidPlatform;
import com.chaquo.python.PyObject;
import com.chaquo.python.Python;

public class MainActivity extends AppCompatActivity {

private EditText serverIp, port, duration, payload;
private Button runServer, runClient;
private LinearLayout clientFields;
private TextView resultView; // Поле для відображення
результату

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

serverIp = findViewById(R.id.serverIp);
port = findViewById(R.id.port);
duration = findViewById(R.id.duration);
payload = findViewById(R.id.payload);

runServer = findViewById(R.id.runServer);
runClient = findViewById(R.id.runClient);

clientFields = findViewById(R.id.clientFields);
resultView = findViewById(R.id.resultView); // Ініціалізація
текстового поля
```

```
// Запуск сервера
runServer.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
String portValue = port.getText().toString();
if (portValue.isEmpty()) {
showToast(getResources().getString(R.string.empty_port));
return;
}

executePythonScript("server", portValue, null, null, null);
}
});
// Запуск клієнта
runClient.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
String ipValue = serverIp.getText().toString();
String portValue = port.getText().toString();
String durationValue = duration.getText().toString();
String payloadValue = payload.getText().toString();
if (ipValue.isEmpty()) {
showToast(getResources().getString(R.string.empty_IP));
return;
}
if (portValue.isEmpty()) {
showToast(getResources().getString(R.string.port_required))
;
return;
}
if (durationValue.isEmpty()) {
durationValue = "10"; // Значення за замовчуванням
}
if (payloadValue.isEmpty()) {
payloadValue = "1400"; // Значення за замовчуванням
}
executePythonScript("client",      portValue,      ipValue,
durationValue, payloadValue);
}
});
}
private void executePythonScript(String mode, String port,
String ip, String duration, String payload) {
if (!Python.isStarted()) {
```

```
Python.start(new AndroidPlatform(this));
}
Python py = Python.getInstance();
PyObject pyScript = py.getModule("run"); // Ім'я файлу
run.py
try {
PyObject result;
if ("server".equals(mode)) {
result = pyScript.callAttr("run_server",
Integer.parseInt(port), false);
} else {
result = pyScript.callAttr("run_client", ip,
Integer.parseInt(port),
Integer.parseInt(duration), Integer.parseInt(payload));
}
if (result != null) {
displayResult(result.toString());
} else {
displayResult(getResources().getString(R.string.empty_resul
t));
}
} catch (Exception e) {
showToast(R.string.error_msg + e.getMessage());
}
}
private void showToast(String message) {
Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
}
private void displayResult(String message) {
resultView.setText(message); // Виведення результату у
TextView
}
}
```

2.2 Текст програми activity_main.xml

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <EditText
        android:id="@+id/port"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/port_txt"
        android:inputType="number"                          />

    <LinearLayout
        android:id="@+id/clientFields"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:visibility="visible">

        <EditText
            android:id="@+id/serverIp"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="IP-адреса                  сервера"
            android:inputType="text"                        />

        <EditText
            android:id="@+id/duration"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/duration_text"
            android:inputType="number"                      />

        <EditText
            android:id="@+id/payload"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/payload_size_text"
            android:inputType="number"                      />
    </LinearLayout>

    <Button
        android:id="@+id/runServer"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/start_server"                 />

    <Button
        android:id="@+id/runClient"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/start_client"                 />
```

```
    <TextView
        android:id="@+id/resultView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/result_field"
        android:textSize="16sp"
        android:paddingTop="16dp"                                    />
</LinearLayout>
```