Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут неперервної освіти
Кафедра управління професійною освітою

ДИПЛОМНА РОБОТА

здобувача другого (магістерського) ступеня вищої освіти

на тему:

«СТРУКТУРА ТА ПРИНЦИПИ СТВОРЕННЯ ХМАРНИХ СХОВИЩ ДАНИХ»

Виконав:
магістрант спеціальності 011 «Освітні, педагогічні науки» (Інформаційно-комунікаційні технології в освіті)
Балан Дмитро Олегович
Науковий керівник:
доктор педагогічних наук,
професор
Войтович Ігор Станіславович

Національна шкала_____
Кількість балів _____Оцінка ECTS____

Члени комісії_____ _____

_____ _____

_____ _____

Київ-2020

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API – Application Programming Interface

DNS – Domain Name System

FTP – File Transfer Protocol

HTML – HyperText Markup Language

HTTP – Hyper Text Transfer Protocol

IEEE – Institute of Electrical and Electronics Engineers

IETF – Internet Engineering Task Force

IP – Internet Protocol

NAS – Network Attached Storage

NFS – Network File System

SQL – Structured Query Language

TCP – Transmission Control Protocol

VPN – Virtual Private Network

ІТ – Інформаційні технології

ОС – Операційна система

СУБД – Система управління базами даних

ХСД – Хмарне сховище даних

ВСТУП

Актуальність теми. В сучасних умовах реалізації усіх інформаційних процесів у глобальному інформаційному просторі, значної актуальності набувають аспекти взаємодії працівників транснаціональних корпорацій, що розміщуються на віддалених територіях і у різних країнах, які мають потреби у вільному доступі до корпоративної інформаційної інфраструктури. Якраз, хмарні сховища даних дають можливість істотно підвищити доступ до даних та потрібних мережних сервісів для їх передавання, опрацювання і зберігання. Географічна віддаленість клієнтів та працівників, потреба у збереженні цілісності даних породжують цілий ряд протиріч, що полягають у доцільності збільшення обсягів розподілених сховищ даних в умовах дотримання вимог щодо їх цілодобової доступності та санкціонованого доступу і захисту цих даних. Тому завдання дослідження структури та принципів створення хмарних сховищ даних є актуальним і своєчасним.

Поряд із цим, існує ряд інших проблем та потреб користувачів, які пов’язані із організацією доступу до хмарних сховищ даних:

– недостатньо розвинена теоретична база дослідження, яка прийшла би на зміну класичній теорії масового обслуговування при проектуванні сучасних систем розподілу даних;

– недостатньо вивчені показники якості функціонування таких систем;

– недостатньо розвинуті самі мережі, які забезпечували б високу якість обслуговування користувачів (працівників) та клієнтів компаній.

Мета і завдання дослідження. Метою дипломної роботи є вивчення структури та принципів передавання даних через хмарні сховища.

Мета магістерської роботи визначає необхідність розв’язання таких завдань:

1. Аналіз стану запровадження та особливостей функціонування хмарних сховищ даних;
2. Розробка моделі хмарного сховища даних;

3. Розробка методу мультипротокольного доступу до даних, що зберігаються у хмарному сховищі;

Об'єкт дослідження – процес передавання даних через хмарні сховища.

Предмет дослідження – методи та засоби організації хмарних сховищ даних.

Методи дослідження. Дослідження, виконані під час роботи над магістерською роботою, ґрунтуються на методах аналізу функціонування хмарних сховищ даних та вдосконалення чинних підходів до організації таких сховищ; комп'ютерного моделювання – для проведення дослідів із протоколами передавання даних, спостереження, синтезу; статистичних – для опрацювання результатів експериментів; проектування – для розроблення елементів архітектури системи хмарного сховища даних.

Наукова новизна здобутих результатів:

1. Вперше описано метод мультипротокольного передавання даних у розподіленій системі хмарних сховищ;

2. Удосконалено модель хмарного сховища даних на основі мультипротокольних засобів.

3. Набув подальшого розвитку метод спільного доступу до сервісів хмарного сховища даних.

Практичне значення здобутих результатів:

– описано уніфікований протокол передавання даних, що дає можливість збільшити пропускну спроможність хмарних сховищ даних;

– описано метод мультипротокольного передавання даних через наскрізний канал передавання даних;

– розроблено метод перерозподілу навантаження на декількох джерелах даних;

– удосконалено елементи архітектури хмарних сховищ даних;

– на основі розробленої архітектури побудовано модель хмарного сховища даних для практичного використання.

Структура та обсяг роботи. Дипломна робота складається зі вступу, трьох розділів, висновків, що містять основні результати роботи, списку

використаної літератури.

РОЗДІЛ 1.
АНАЛІЗ РОЗВИТКУ ХМАРНИХ СИСТЕМ ЗБЕРІГАННЯ ДАНИХ

1.1. Основні підходи до проєктування хмарних технологій

Хмарні обчислення (cloud computing) – сучасна Інтернет-технологія, яка забезпечує користувачів доступом до спільних ресурсів. Користувач отримує доступ до своїх даних, але не формує ні інфраструктуру, не визначає ні операційну систему, ні програмне забезпечення, з яким йому доводиться працювати. До таких технологій, зокрема, відносяться й сховища даних, різні Інтернет-сервіси, загальне та спеціальне програмне забезпечення.

Термін «хмара» використовується в переносному значенні, як образ певної інфраструктури, які поєднують апаратне і програмне забезпечення, яке втім недоступне для користувача.

Згідно документу IEEE (2008 р.), «Хмарна обробка даних – це філософська парадигма, в рамках якої дані постійно зберігаються на серверах в мережі Інтернет і тимчасово зберігаються (кешуются) на пристрої клієнта (персональних комп’ютерах, ігрових приставках, ноутбуках, смартфонах і т.д.)» [10, 16, 21].

За оцінками аналітиків, ринок хмарних обчислень за останні 10 років із $ 17 млрд – зріс до $ 83 млрд. Крім того, понад 30 % компаній у всьому світі вже почали використовувати хоча б один хмарний сервіс.

Основними причинами доступності хмарних систем є:

1. Розвиток багатоядерних процесорів та серверів на їх основі:
– значне збільшення продуктивності;
– зниження їхньої вартості;
– зниження енергоспоживання хмарних систем.
2. Збільшення обсягів носіїв даних, що в свою чергу дало змогу:
– збільшити об'єми збережуваних даних;
– зменшити собівартість обслуговування сховищ даних.
3. Розвиток технології мультипоточного програмування привело до:

– ефективного розподілу обчислювальних ресурсів багатопроцесорних систем;

– гнучкого розподілу потужностей та ресурсів у хмарі.

4. Розвиток технологій віддаленого доступу привело до:

– створення хмарного програмного забезпечення;

– легкість масштабування і нарощування як системи, так і сервісів;

– зменшення витрат на обслуговування хмарних систем;

– доступність через Інтернет.

5. Збільшення пропускної здатності мереж привело до:

– збільшення швидкості роботи з хмарними сервісами;

– зниження вартості доступу до Інтернет;

– поширення хмарних обчислень серед багатьох користувачів.

Усі перераховані чинники привели до підвищення ролі хмарних обчислень в IT галузі. У їх основі лежить декілька підходів:

1. Доступність через Інтернет.

2. Віртуалізація.

3. Хмарні обчислення – це перш за все послуга без деталізації про те, що ж використовується для її надання.

4. Простота і відповідність стандартам. Хоча хмари і є найновішими розробками у сфері IT, проте вони не потребують і не використовують якихось нових мов програмування, створення та налаштування конфігураційних файлів чи необхідності довготривалого їх налаштування.

Інститут інженерів з електротехніки та електроніки (США) при характеристиці хмарних обчислень використовує принцип «4-3-2». Перша частина визначає обов’язкові характеристики хмарних сервісів [11]:

– Самообслуговування на вимогу клієнта (self service on demand), користувач сам визначає і змінює свої потреби без прямого контакту із постачальником хмарних послуг;

– Поєднання ресурсів (resource pooling), постачальник послуг об’єднує ресурси і надає їх для обслуговування достатньо великої кількості споживачів в єдиний пул для перерозподілу їх між ними в умовах постійної зміни запиту від

них на необхідні потужності;

– Еластичність (elastic), означає, що ці послуги можуть бути надані, а за потреби збільшені чи зменшені у будь-який момент часу, без додаткових запитів;

– Облік споживання послуг (usage based) здійснюється автоматично.

З точки зору надавача послуг, хмарні сервіси та сховища даних дають змогу використовувати менше апаратних ресурсів, ніж при виділенні системних ресурсів для кожного користувача за рахунок автоматизації виділення ресурсів сервера внаслідок чого значно знижуються витрати на їхнє обслуговування.

З точки зору користувача хмарних сервісів та сховищ даних, це дає змогу отримувати послуги з вищим рівнем доступності (high availability) і меншими ризиками недієздатності, забезпечити швидке масштабування інформаційної системи завдяки еластичності без потреби створювати, обслуговувати і постійно удосконалювати апаратну інфраструктуру [3].

Згідно принципу «4-3-2», друга цифра характеризує основні методи постачання хмарних сервісів: Infrastructure-As-A-Service, Platform-As-A-Service і Software-As-A-Service (рис. 1.1):

– IaaS – набір пов’язаних із інфраструктурою можливостей та функціоналу (операційна система, мережеві підключення і т.д.), які надаються користувачу на основі оплати за використання і використовуються для розміщення додатків. Найбільш поширені продукти цього сегменту від компаній Amazon, Microsoft, VMWare, Rackspace та Red Hat.

– PaaS – функціональність вищого рівня (порівняно з IaaS), що пов’язана із платформою і надається як сервіс для розробників додатків. З PaaS розробники абстрагуються від нижчої інфраструктури. Наприклад, Google Apps – застосунки для бізнесу в режимі онлайн, доступ до яких відбувається за допомогою Інтернет-браузера тоді як програмне забезпечення і дані зберігаються на серверах Google [15].

– SaaS – програмне забезпечення, яке пропонується в якості Інтернет-сервісів, коли організації просто споживають і використовують додатки. Прикладами програмного забезпечення як послуги, що працює на основі обчислювальної хмари, є сервіси Google Docs [12].

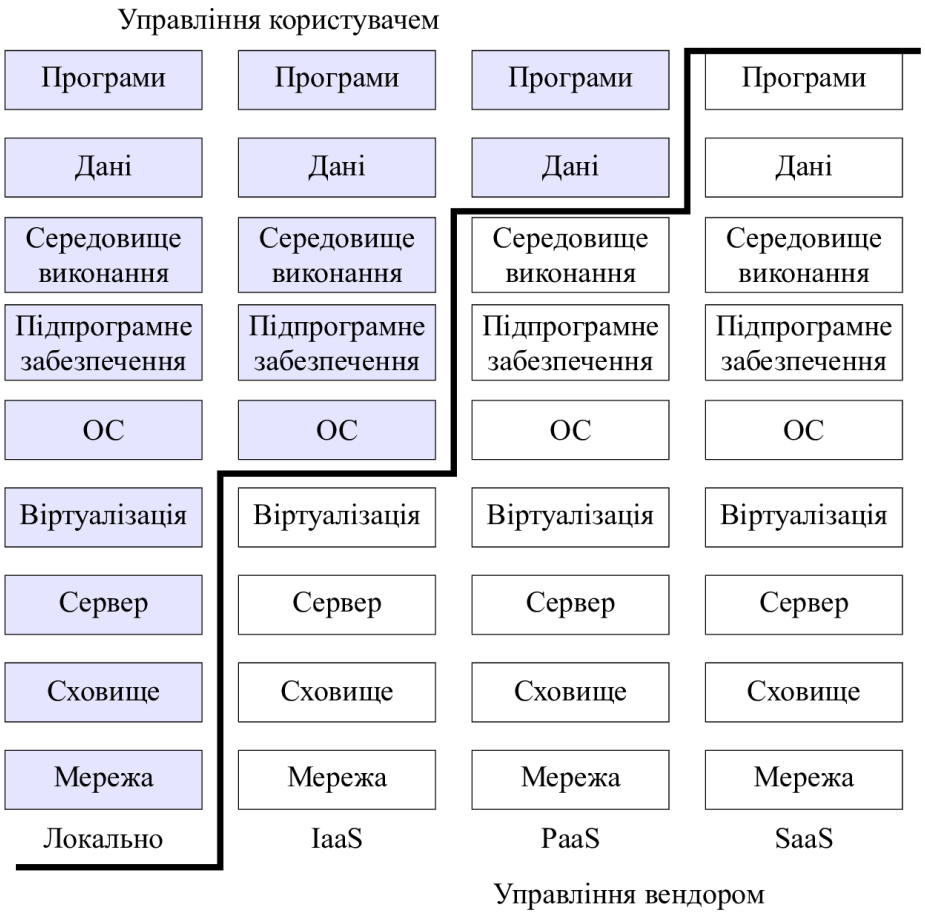


Рис. 1.1. Рівневе представлення хмарних сервісів.

Зі зростанням рівня постачання хмарних послуг зростає й цінність для кінцевого користувача, і, відповідно, їх кількості (рис. 1.2). Ці три види хмарних сервісів можуть використовуватись як окремо так і в комбінації один з одним.

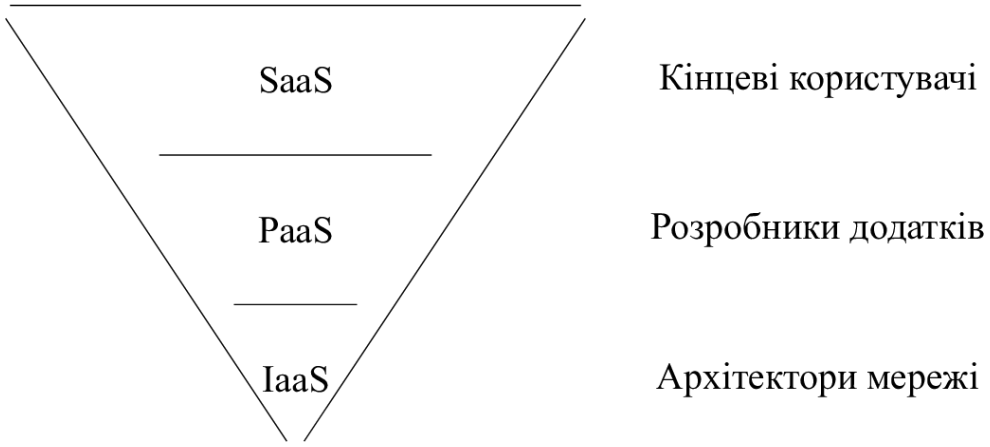


Рис. 1.2. Моделі роботи з хмаркою для різних груп користувачів.

Остання цифра в принципі «4-3-2» характеризує тип хмари: приватний чи публічний. Для користувача використання сервісу, який розміщений в приватній хмарі чи розміщений в публічній, може зовсім не відрізнятись.

Таким чином, хмарна технологія охоплює окрім самих хмарних обчислень, такі сфери, як зберігання даних, інтеграцію сервісів, організацію процесів управління.

До переваг концепції використання хмарних сервісів провайдерами відносять:

– розвиток додатків проміжного рівня (middleware).

– зниження вартості і підвищення масштабованості інформаційних ресурсів.

– переваги послуг, доступ до яких надається через «мережеву хмару» в порівнянні з іншими видами Інтернет-послуг.

– хмарні обчислення створюють платформу для нового виду Інтернет-послуг.

До переваг концепції використання хмарних сервісів замовниками відносять:

– зниження витрат, пов’язаних із придбанням апаратних і програмних засобів.

– зменшення вартості (у випадках оплати за фактом використання).

– зручне придбання нових послуг і прискорення виходу на ринок.

Цей огляд показав особливості та підходи до використання хмарних сервісів та сховищ даних з точки зору користувача. Потрібно також вивчити підходи до архітектури хмарних технологій.

1.2. Аналіз архітектур хмарних сховищ даних.

Хмарна система опрацювання та зберігання даних – це умовний термін, який означає такий спосіб зберігання даних, який можна адмініструвати віддалено через спеціальний користувацький інтерфейс, який надає кожному користувачу права адміністратора на його дані. Такий інтерфейс спеціально розроблено для того, щоб не залежати від місцезнаходження самої системи та сховища даних, так, що не має значення де вона знаходиться: на локальному

комп’ютері користувача, на сервері провайдера, або компанії, що надає відповідну послугу. Хмарні структури зберігання та опрацювання даних утворюють принципово нові підходи до архітектури, що в свою чергу підтримують різні рівні доступу до обслуговування достатньо великої географічно розподіленої групи користувачів і накопичувачів даних [4].

В такій ситуації спроможність клієнта контролювати і керувати тим, як і де зберігаються його дані має дуже важливе значення як для нього самого, так і для компанії, яка надає відповідні послуги і використовує при цьому різні механізми монетизації. Зазвичай, постачальники хмарних послуг надають користувачам та засоби управління, які б забезпечували користувачам надійний контроль над їхніми витратами.

Ефективність та цілісність зберігання даних у хмарних сховищах – важлива характеристика хмарної інфраструктури, враховуючи її концепцію до економії як кошщтів, так і ресурсів. Для того, щоб хмарна система зберігання даних була ефективнішою, потрібно забезпечити можливість зберігати якомога більше даних. Вдалим рішенням у такому випадку є скорочення обсягів даних, щоб вони займали якомога менше фізичного простору на серверах. Виділяємо два основні способи досягнення цього:

1) стиснення або упаковка даних шляхом їх перекодування з використанням різних способів представлень;
2) виключення дублікатів даних, які могли б містити ту саму інформацію просто збережену з різними часовими інтервалами.

Хоча обидва методи бувають досить ефективними, проте є певні додаткові навантаження на систему, оскільки стиснення передбачає перекодування даних, а виключення дублікатів даних передбачає обчислення правил для пошуку дублікатів.

При розгляді архітектури хмарного сховища даних необхідно враховувати її параметри. Такими основними параметрами хмарних сховищ даних є характеристики архітектури: вартість, швидкість доступу, забезпечення віддаленого доступу, продуктивність тощо (рис. 1.3).

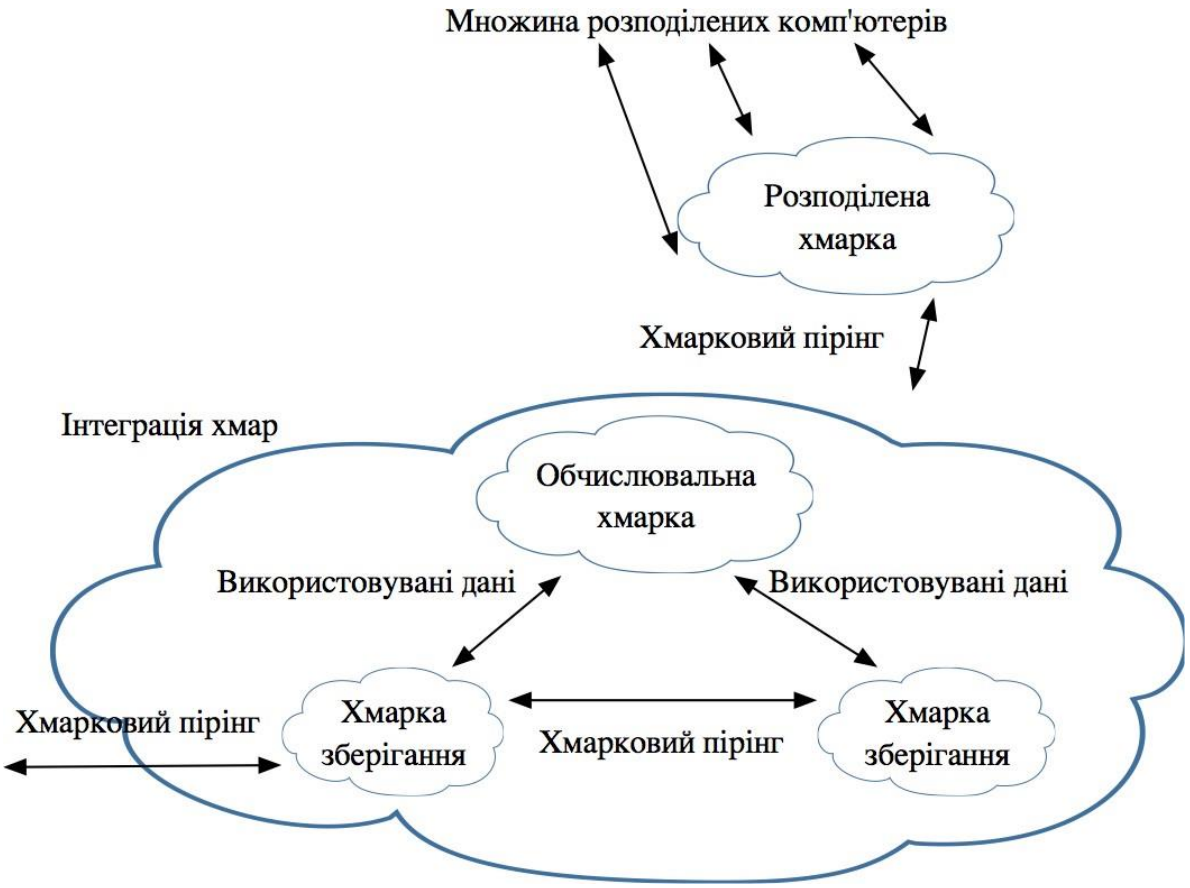


Рис. 1.3. Моделі хмарної екології.

Архітектура хмарного сховища зберігання даних – це, насамперед, доступ до ресурсів зберігання даних за вимогою користувача в масштабованому середовищі. Загалом архітектура хмарного сховища даних являє собою інтерфейс, який надає компанія користувачу для доступу до своїх накопичувачів даних (рис. 1.4).

У попередніх версіях систем зберігання даних на серверах здійснювалось за протоколом SCSI, але в хмарній архітектурі з’являються нові протоколи. Серед них є як зовнішні протоколи веб-сервісів і файлові протоколи так і традиційні інтерфейси (SCSI, iSCSI та ін.). За зовнішнім інтерфейсом знаходиться наступний рівень програмного забезпечення, що відповідає за логіку зберігання даних. Тут реалізовано цілий ряд необхідних функцій: реплікація даних; скорочення обсягу даних; визначення оптимального розміщення даних з урахуванням геолокації сервера та користувача. Внутрішній інтерфейс забезпечує власне фізичне зберігання даних на сервері з фізичними дисками. На основі такої архітектури виокремлено її характеристики (табл. 1.1).

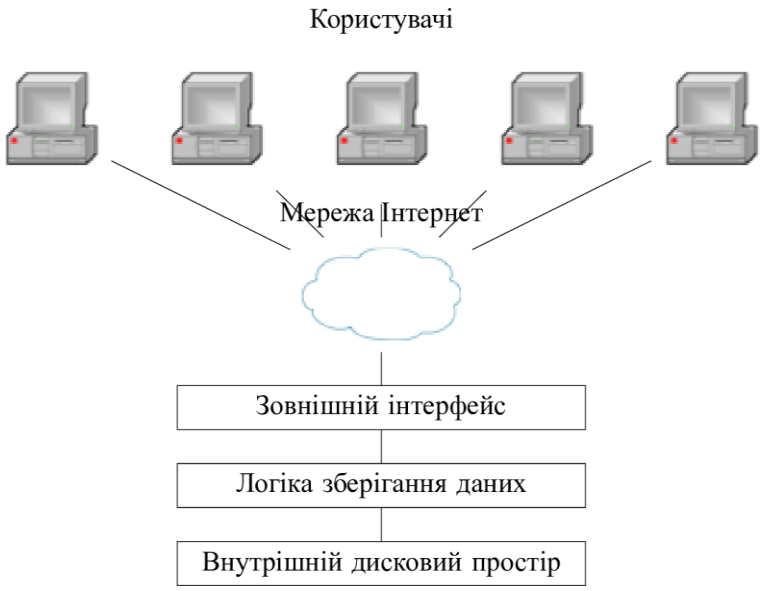


Рис. 1.4. Архітектура хмарного сховища зберігання даних.

Однією з найбільших відмінностей між хмарною та традиційною системами зберігання даних є власне засоби доступу до них (рис. 1.5). Багато постачальників таких послуг пропонують різні методи доступу, однак найбільшприйнятим є API веб-сервіс. Більшість з них реалізовані на такому принципі, основою якого є об’єктно-орієнтований підхід, що реалізований на основі протоколу HTTP (з використанням TCP).

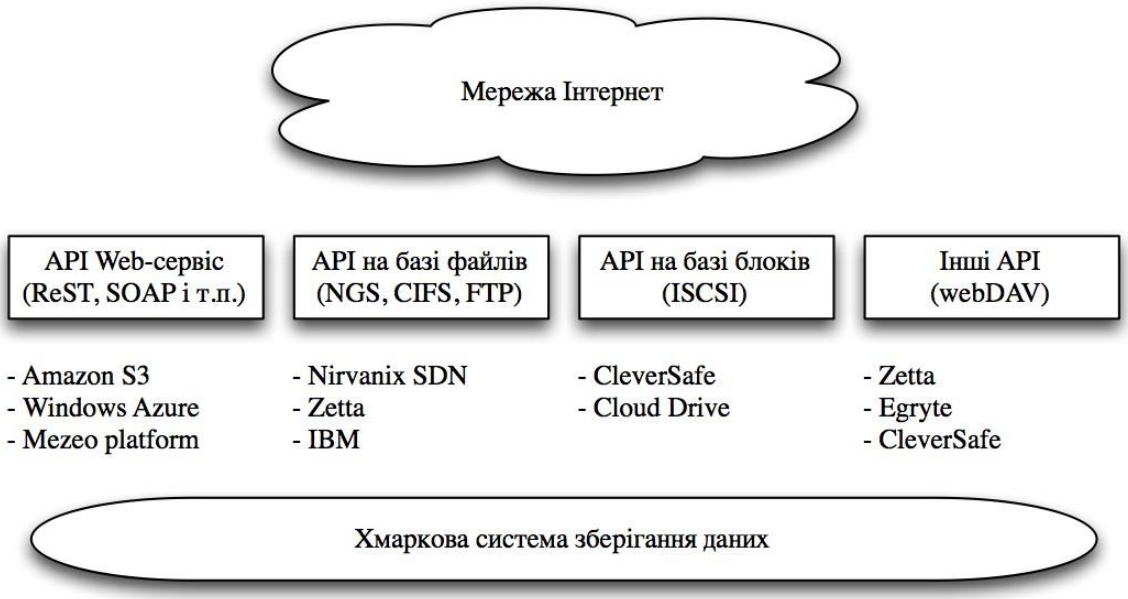


Рис. 1.5. Методи доступу до хмарних систем зберігання даних.

Таблиця 1.1.

Характеристики архітектури хмарного сховища зберігання даних

Характеристика	**Опис**
Керованість	Здатність користувача керувати системою
Метод доступу	Протокол надання хмарного зберігання даних
Продуктивність	Пропускна здатність Час затримки
Мультикористування	Одночасна підтримка багатьох користувачів
Масштабованість	Передбачена можливість нарощування з метою задоволення нових вимог і запитів, або ж обробки високого навантаження на систему
Управління	Гнучке управління системою: вибір вартості, продуктивності, тощо
Ефективність	Завантаженість та використання накопичувачів
Вартість	Середня вартість зберігання даних

Одна з проблем застосування API веб-сервісів виникла через те, що для використання усіх переваг хмарної системи зберігання даних, вони вимагають постійної інтеграції з відповідним додатком, у якому вони опрацьовуються. Тому з хмарними системами зберігання даних для забезпечення такої інтеграції використовуються загальні методи доступу до даних: це файлові протоколи (NFS / Common Internet File System (CIFS), FTP), або ж блочні протоколи (iSCSI). Вищезазначені протоколи найбільш поширені, проте для хмарного зберігання даних підходять й інші. Можна зустріти й такі компанії, що підтримують декілька протоколів доступу до даних у хмарі. Наприклад, IBM дає змогу одночасно використовувати файлові протоколи (NFS і CIFS) а також протоколи на основі SAN в одній інфраструктурі хмарної системи зберігання даних. Однак,

при такому використанні різних протоколів обміну даними між користувачами та хмарними сховищами даних виникає багато збоїв та перешкод для забезпечення оптимального надання доступу до них. Переважно, ці проблеми пов’язані з недостатньою пристосованістю мережі Інтернет до надання саме таких послуг.

Основним завданням хмарної системи зберігання даних є швидке переміщення даних від користувача до віддаленого постачальника хмарних послуг і назад. Основна проблема тут криється в протоколі TCP, який є основним протоколом Інтернет. Адже саме TCP управляє потоком даних на основі принципу підтвердження прийому пакету даних із віддаленого вузла - сервера. Втрата чи затримка окремих пакетів обмежують продуктивність цих послуг для уникнення мережних проблем. TCP чудово підходить для переміщення незначних обсягів даних через мережу Інтернет, але аж ніяк не для доставки великих об’ємів даних, адже у такому випадку час обміну даними значно збільшується, а продуктивність відповідно падає.

Для усунення цього недоліку створено новий протокол Fast and Secure Protocol (FASP), який було розроблено для пришвидшення передачі даних в умовах достатньо великого часу відгуку і можливої втрати пакетів даних. Основою для нього служить протокол UDP, що є допоміжним протоколом TCP. Таким чином, UDP дає змогу керувати цими проблемами, передаючи цю складову до протоколу FASP (рис. 1.6).

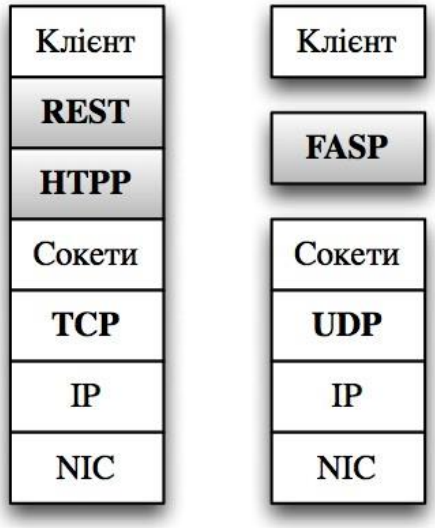


Рис. 1.6. Архітектура протоколу Fast and Secure Protocol.

Працюючи на стандартних мережних адаптерах, FASP ефективно використовує доступну смугу пропускання і оминає проблемні місця традиційних моделей масової передачі даних.

Однією з ключових особливостей побудови хмарного сховища є його багатокористувацький режим, адже хмарне сховище даних використовується одночасно багатьма користувачами. Ця обставина відноситься до різних рівнів хмарної системи зберігання даних – від рівня додатку, де користувачу виділяється обмежений простір імен, до рівня основного зберігання, де користувачам виділяються окремі фізичні чи віртуальні накопичувачі. Багатокористувацький режим поширюється і на мережну інфраструктуру, яка забезпечує доступ користувачів до накопичувачів, забезпечуючи стабільність та якість обслуговування і окрему смугу для передавання даних до конкретного користувача.

Масштабованість розглядають з кількох точок зору, але щодо хмарних сховищ даних ця властивість розглядається як можливість виділення ресурсів для зберігання на вимогу. Можливість реалізації такої динамічної зміни обсягів ресурсів для зберігання даних (як збільшення, так і зменшення) гарантує економічну ефективність для користувача і додаткову трудність для постачальника хмарних послуг. Адже масштабованість повинна забезпечуватися як для системи зберігання даних, так і для забезпечення її пропускної спроможності, а також завдяки концепції побудови хмарних сховищ даних, система передбачає їх розташування максимально близько до користувача саме завдяки перерозподілу ресурсів для хмарного зберігання даних. Якщо доступ користувач отримує в режимі «тільки для читання», то реалізовуються також механізми реплікації і розповсюдження даних (рис. 1.7).

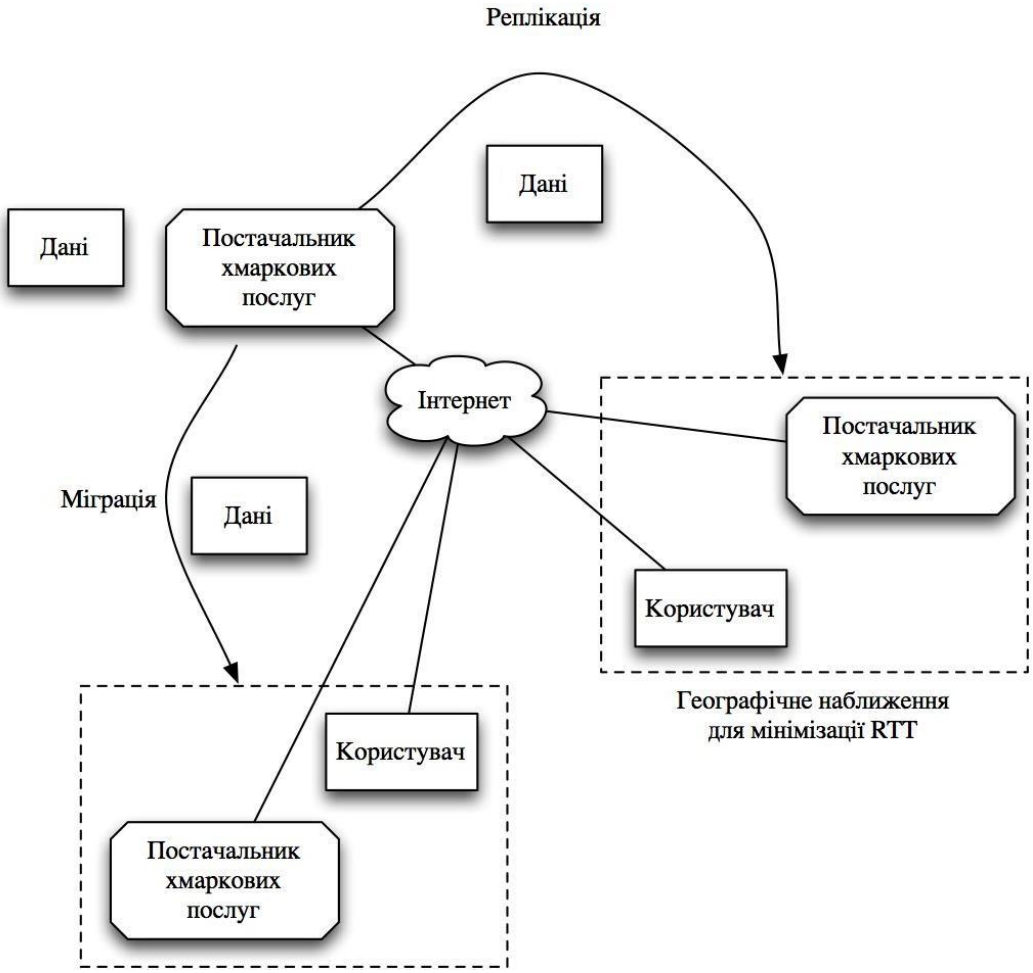


Рис. 1.7. Масштабованість хмарної системи зберігання даних [18].

Системи хмарного зберігання даних розвиваються в трьох основних напрямках, один з яких допускає поєднання двох інших для досягнення ефективності та безпеки даних [7].

Постачальники загальнодоступних (публічних) хмарних систем зберігання даних надають свою інфраструктуру на умовах оренди. Приватні хмари побудовані на тих самих концепціях, що й публічні, але їхня інфраструктура інтегрована в приватну мережу користувачів. Гібридні хмарні системи зберігання даних забезпечуюьб поєднання обох моделей, регулючи, які дані зберігати в приватній хмарі, а які у публічних хмарах (рис. 1.8).

Рис. 1.8. Хмарні моделі зберігання даних [18].

Таким чином, гібридні моделі хмар дають змогу організаціям забезпечувати конфіденційність своїх даних у межах їх власних серверів, передаючи відкриті дані в публічну хмару. Залишається лише перенести використовувані організацією додатки в хмару, або ж перейти на нові хмарні сервісі, згадані вище. Це не завжди можливо зробити через особливості конкретного додатка, його взаємодії із іншими системами або сервісами, які не будуть перенесені в хмару. І, навіть, коли перенесення теоретично можливе, однак це може вимагати або значної переробки програмного коду, або переписування всього програмного забезпечення з нуля, що може знизити економічну ефективність такого переходу і зрештою, від нього можуть віжмовитись узагалі. Адже, є багато таких додатків, які просто неможливо перенести у хмару: відеоредактори, CAD-системи, програми реального часу. Втім, останнім часом для розробників та досвідчених користувачів створюється новий набір API, пропонуються нові платформи для програмування та адаптації вже розроблених додатків, розроблено нові інструменти для оптимізації роботи

з хмарними версіями програмних додатків.

Також необхідно забезпечувати постійне підключення до інтернет, що в наших реаліях може бути визначальною перешкодою для впровадження хмарних сховищ зберігання та опрацювання даних. Частково, це реалізовано за рахунок того, що з окремим сервісами можна працювати в оффлайн-режимі, а при підключенні до сервера все синхронізувати [30].

1.3. Аналіз рівнів транспортування даних в розподілених системах.

Збільшення пропускної спроможності мереж до 10 Гбіт/с, а іноді й до 40 Гбіт/с дає змогу транспортувати дані як користувачам, так і програмам на відносно стабільному рівні. Такі високошвидкісні мережі з’явилися завдяки оптоволоконним лініями зв’язку [27]. Між тим, обсяги даних, що передаються в цих мережах також зростають експоненціально, тому існуючі способи зберігання та передавання даних вже неефективний в сучасних умовах.

З цією метою розроблені та впроваджуються високопродуктивні серверні системи і мережі значно інтенсифікували розвиток розподілених обчислень. Побудовані за такими принципами, хмарні розподілені додатки спершу характеризувалися за вимогами до швидкості зв’язку, однак на сьогодні, такі додатки та хмарні сервіси потребують стабільного доступу та високою пропускної спроможності мережі. Ці нові вимоги частково реалізовуються на рівні застосунків та мереж, проте необхідно здійснювати оновлення та адаптацію транспортного рівня.

Нині навіть високошвидкісні мережі не можуть забезпечити оптимального використання програмним забезпеченням, оскільки протокол управління передачею даних (TCP), який залишається основним транспортним протоколом Інтернет, недостатньо використовує пропускні спроможності мережі навіть при високошвидкісних з’єднаннях [14, 19, 28]. Адже , один потік TCP з базовми налаштуваннями параметрів за замовчуванням на Linux 2.4 може досягти максимальної швидкості лише до 5 Мб/с, тоді як він буде працювати навіть на каналі зі швидкістю 1 Гбіт/с.

Це зумовлено відсутністю зв’язку між мережним рівнем і транспортним

рівнем, що призвело до неспроможності серверів ефективно обслуговувати потреби користувачів та їх додатків і даних.

Наприклад, веб-браузер буде працювати однаково, і коли комп'ютер підключений через мережу Ethernet, і через Wi-Fi. Це відбувається тому, що той пласт мережі, що відповідає за обробку доступу до середовища надає ті самі послуги вищому пласту, незалежно від того, дротове чи бездротове середовище використовується для передачі даних на локальному рівні.

Розглядають такі дві моделі мережних пластів:

1) модель ISO взаємодії відкритих систем (OSI) включає такі сім пластів [6, 20]:

а)програма,

б)презентація,

в)сесія,

г)транспорт,

д)мережа,

е)передача даних

ж)фізичний рівень.

2) рівнева модель IETF (Internet Engineering Task Force) TCP / IP, що включає такі чотири пласти [6, 13]:

а)додаток,

б)транспорт,

в)мережа,

г)доступ до мережі.

Модель OSI застосовується переважно в академічному середовищі, тоді як модель IETF стала основним набором протоколів, на основі яких функціонує Інтернет.

Розглядаючи рівневу архітектуру TCP/IP, у ній також, як і в архітектурі OSI, можна виділити рівні, функції яких забезпечують технологічну реалізацію мережі, а також рівні, які орієнтовані лише на роботу з додатками.

З цією метою у стеку TCP/IP визначені такі чотири рівні:

– прикладний рівень протоколу,

– основний (транспортний) рівень протоколу,

– рівень міжмережних інтерфейсів,

– рівень мережних взаємодій.

Кожен із них несе в собі певне навантаження для виконання основного завдання: організації надійної і продуктивної роботи комп'ютерної мережі, побудованої на основі різних мережних технологій, моделей та протоколів.

Стрижнем архітектури моделі IETF є рівень міжмережної взаємодії, або мережний рівень, що забезпечує передавання пакетів в режимі без встановлення прямих з'єднань. Саме цей рівень і забезпечує можливість переміщення пакетів даних мережею, будуючи найбільш раціональний маршрут. Цей рівень ще називають рівнем Інтернет, акцентуючись на його основну функцію – передавання даних через мережу.

Разом з тим, протоколи двох нижніх рівнів є мережно-залежними, а відповідно, модулі протоколів міжмережного рівня і рівня мережних інтерфейсів встановлюються і на кінцевих вузлах мережі, і на проміжних маршрутизаторах.

У таблиці 1.2. показано відповідність рівнів стеку TCP/IP рівням моделі OSI.

Таблиця 1.2.

Відповідність рівнів стеку TCP/IP і моделі OSI

OSI							TCP/IP
7	WWW	SNMP	FTP	Telnet	TFTP	SMTP	I
6	Gopher, WAIS						
5		TCP			UDP		II
4							
3	IP	ICMP	RIP	OSRF		ARP	III
2	не регламентується: Ethernet, Gigabit Ethernet,						IV
1	Token Ring, PPP, FDDI, X.25, SLIP, frame relay …						

Тобто транспортний рівень мережі може забезпечити стабільне передавання даних пакетами навіть через мережу з низькою надійністю. З цією

метою, транспортний рівень розділяє потік даних на окремі пакети і, використовуючи мережний рівень, здійснює їх доставку до кінцевої точки, де він вже відповідає за збір пакетів з мережного рівня і компонування їх у потік даних. Мережний рівень зберігає таблицю маршрутизації, у якій вказується, який саме вузол повинен використовуватися для кожного призначення, і переключає всі вхідні пакети в заданому напрямку. Зрозуміло, що кожен пласт має доступ до сусідніх пластів зверху і знизу завдяки стандартним інтерфейсам, а всередині пластів забезпечено доступ до гнучкіших способів передачі даних.

У рівневій архітектурі транспортний рівень є четвертим пластом, що знаходиться між мережним і прикладним рівнями (рис. 1.9). Існують декілька транспортних протоколів на цьому рівні:

-TCP,

-UDP,

-SCTP,

-DCCP.

TCP залишаєтьсяє самим надійним протоколом передавання потокових даних, тоді як UDP ненадійний протокол обміну інформацією.

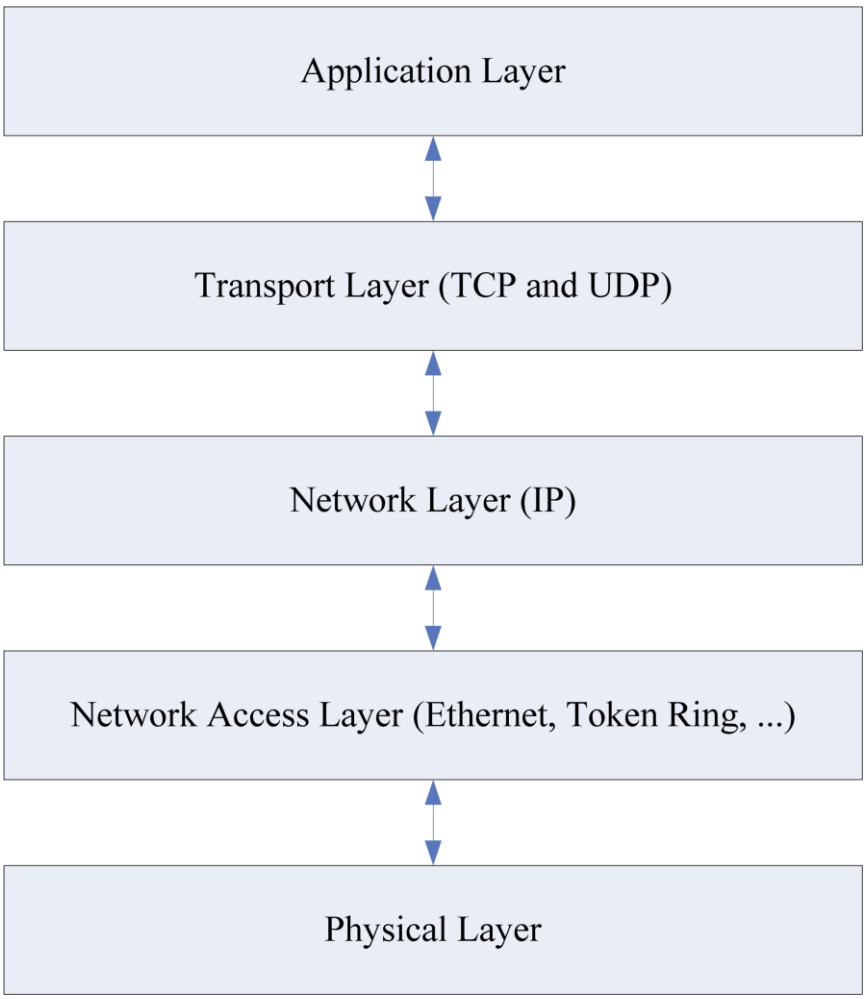


Рис. 1.9. Рівні протоколів передачі даних [6].

Поступовий розвиток двох крайніх пластів протоколу із незмінністю протоколу TCP/IP свідчить про неможливість повної адаптації нових сервісів між додатком і фізичним рівнем.

Сучасний стан пропускної спроможності протоколів можна зобразити, як пісочний годинник, що демонструє вузькі місця протоколів (рис. 1.10).

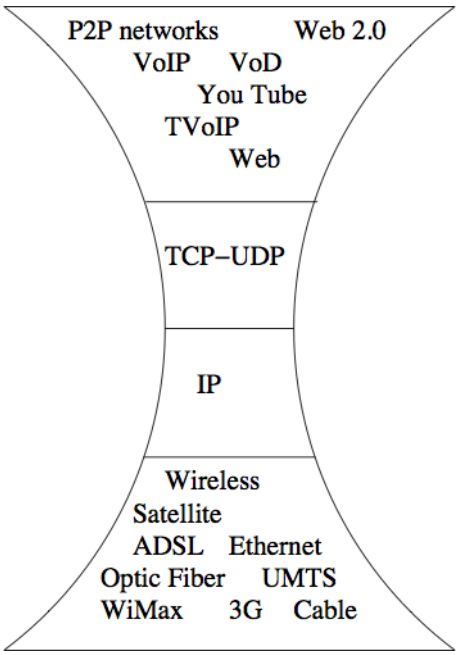


Рис. 1.10. Пропускна спроможність Інтернет-протоколів [6].

За [26], близько 95 % всіх переданих даних в Інтернет передаються завдяки протоколу TCP, який забезпечує надійне передавання даних через нестабільне середовище передачі. Для забезпечення ефективності роботи протокол TCP повинен виконувати такі основні завдання:

- досягнути максимальної смуги пропускання,
- відновлювати швидкість передачі даних до максимальної після її падіння через перевантаження, або втрату пакетів.
- залишатися в стабільному стані, поки не зміниться стан мережі.

Швидкість передавання даних визначається або за часом між переданими/прийнятими пакетами або їх кількістю за одиницю часу.

Отже, основним завданням передавання даних у мережі є подолання вузьких місць, для чого розроблено механізм управління перевантаженням в стеку протоколів TCP/IP, що забезпечує стабільшіший і швидший зв'язок, враховуючи технічні паратери і характеристики пристроїв та засобів комунікації.

Висновки до розділу 1

У цьому розділі встановлено:

1. Хмарна система зберігання даних – це умовне поняття, яке відповідає віддаленій системі, яка може адмініструватися користувачем через спеціальний інтерфейс.

2. Проаналізовано архітектуру систем зберігання даних, описано підходи до проєктування хмарних технологій

3. Здійснено аналіз проблеми транспортування даних в розподілених системах, в ході чого виділено позитивні і негативні риси різних архітектур зберігання даних.

4. Розглянуто хмарні технології з точки зору їх використання для зберігання та поширення даних.

5. Проведено аналіз архітектур сховищ даних. Здійснено поділ на приватні, публічні та гібридні.

6. Проаналізовано рівні та способи транспортування даних в розподілених системах. Визначено TCP основним протоколом, який використовується для передачі даних.

7. Виявленно слабкі місця в організації та транспортуванні даних до хмарних сховищ даних.

РОЗДІЛ 2.
МОДЕЛЮВАННЯ ХМАРНОГО СЕРЕДОВИЩА ДАНИХ І АНАЛІЗ МЕТОДІВ ЇХ ПЕРЕДАВАННЯ

2.1. Моделі передачі даних в хмарних технологіях.

Першою моделлю передавання даних між користувачами була модель «точка–точка», яка передбачала окремий виділений канал передачі від того абонента, який передавав, до того абонента, який приймав і застосовувалася в телефонії до 90-тих років XX століття. Саме її використано для побудови перших комп'ютерних мереж.

Мережа типу «точка–точка» – це найпростіший вид комп'ютерної мережі, де два персональні комп'ютери з'єднані напряму через спеціальне комунікаційне обладнання (рис. 2.1). Таке з'єднання просте і дешеве, однак з'єднати таким способом можна лише два персональні комп'ютери чи пристрої.

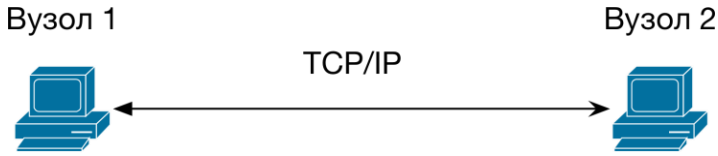


Рис. 2.1. Модель мережі «точка–точка», побудовано за даними [6].

Така модель може використовуватися науковцями та інженерами для представлення передачі даних між двома об'єктами. Для подолання недоліків такого прямого з'єднання типу «точка-точка» запропоновано модель віртуальної приватної мережі (VPN) [2] з кращою якістю обслуговування, для чого у ній було вперше використано зображення хмари для позначення розмежування між користувачем і постачальником Інтернет-послуг.

Модель VPN – це мережа, створена на базі загальнодоступних або віртуальних каналів інших мереж. Безпека передавання пакетів даних через публічні мережі реалізується з використанням алгоритмів шифрування, внаслідок чого канал обміну інформацією стає захищеним. VPN може об'єднати декілька територіально віддалених локальних мереж організації в єдину мережу, використовуючи для зв'язку між ними виділені каналих.

Отже, в загальному VPN складається з двох частин: «внутрішньої» мережі (їх може бути декілька), і «зовнішньої» мережі, через яку й проходять їх з'єднання (зазвичай - це Інтернет) (рис. 2.2).

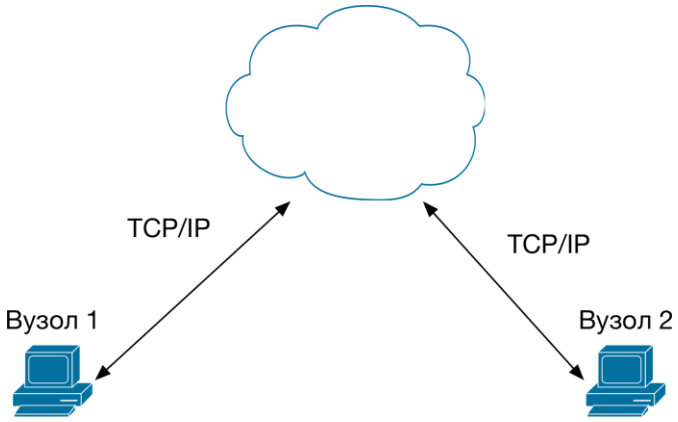


Рис. 2.2. Модель передачі даних через VPN [6].

Підключення до VPN будь-якого користувача здійснюється за допомогою точки доступу, яка підключена як до внутрішньої, так і до зовнішньої мережі. Для цього заазвичай використовуються локальні сервери, а від віддаленого користувача сервер просить ідентифікації та аутентифікації. Після вдалого здійснення обох процесів, відбувається процес авторизації віддаленого користувача (чи віддаленої мережі), внаслідок чого він /вона наділяється повноваженнями для роботи в цій мережі.

При такому поєднанні декількох локальних мереж в загальну VPN мережу створюється спільний простір опрацювання даних при відносно мінімальних витратах і достатньому ступеню захисту. Для створення подібної мережі на одному з комп'ютерів у кожному сегменті встановлюється спеціальний VPN-шлюз, що забезпечує передачу даних між підмережами чи користувачами. Обмін даними в кожному сегменті здійснюється звичайним способом, проте коли необхідно передати дані в інший сегмент VPN-мережі, то вони передаються саме через шлюз, де здійснюється обробка даних, їх шифруванн та передавання загальною мережею (Інтернет) шлюзу в іншому сегменті мережі. Там дані розшифровуються і передаються на кінцевий комп'ютер.

Усе це відбувається зовсім непомітно для кожного користувача і нічим не відрізняється від роботи в звичайній локальній мережі. Також VPN

використовується для надання доступу комп’ютерів користувачів до локальної мережі організації.

До недоліків моделі VPN можна віднести потребу в закупівлі додаткового обладнання і серверного програмного забезпечення, а також необхідність нарощування зовнішнього трафіку. Проте, ці видатки порівняно незначні і враховуючи переваги VPN, вона досить часто використовується для побудови корпоративних мереж організацій для забезпечення спільної роботи з даними.

Розглянуті моделі передачі даних потребують прямої участі пристроїв абонентів у цьому процесі, тобто обладнання має працювати цілодобово, щоб мережа фнукціонувала постійно. Позбавлена цього недоліку модель, спроєктована на основі ідеї виділення так званого дата-центру– центру зберігання та опрацювання даних, спеціалізованого сервера, чи групи серверів, які можуть і не належати цій організації, але підключений у її автономну систему або сегменти мережі через множину каналів зв’язку.

Такий дата-цент являє собою «сукупність приміщень, зі встановленим обладнанням та обслуговуючим персоналом, що утворюють єдиний фізичний простір для розміщення комп’ютерів, електронних та інших засобів прийому, передачі, обробки, зберігання даних і забезпечують задану ступінь доступності, розміщеного обладнання в заданому режимі функціонування» [8].

Створення та промислове використання дата-центрів здійснюється з дотримання стандартів та вимог. Основна фнкціональна діяльність дата-центру виражається у закупівлі та обслуговуванні серверного і комунікаційного обладнання згідно з умовами договрів.

Типова структура дата-центру містить такі основні складові [9]:

1) основна розподілююча підсистема MDA (Main Distribution Area) – забезпечує інтерфейс доступу до дата-центру і здійснює розподіл трафіка головної магістралі на внутрішні магістралі;

2) горизонтальна розподільча підсистема HDA (Horizontal Distribution Area) – керує трафіком внутрішніх магістралей;

3) підсистема розводки на обладнання EDA (Equipment Distribution Area) – доставляє трафік до серверів та дискових масивів.

Модель обміну даними за допомогою дата-центру можна зобразити графічно (рис. 2.3).

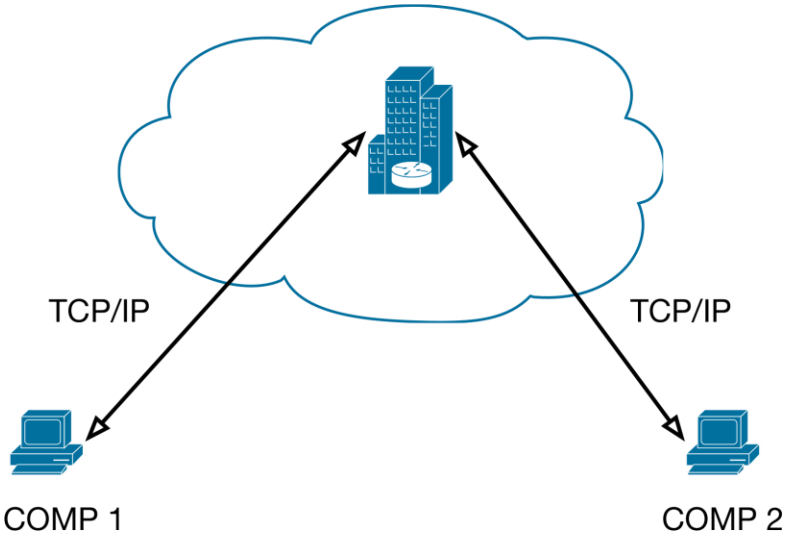


Рис. 2.3. Модель обміну даними через дата-центр [6].

Таке подання моделі дата-центру дає змогу моделювати зовнішні зв’язки центру, і, хоча, не відображає його внутрішньої структури, відображає можливості обміну даними між користувачами. Реальний дата-центр складається із декількох серверів, які можуть знаходитись у одному місці або ж розділені регіонально.

Файли, якими обмінюються користуввачі дата-центру, зберігаються на центральному (головному) сервері (центральних (головних) серверах), а доступ до них відбувається через так звані сервери-сателіти.

Узагальнена модель центрального (головного) сервера містить (рис. 2.4):

1. Firewall – захисний програмний модуль.
2. Nginx – веб-сервер для обробки запитів та переадресування їх на інтерфейсний модуль центрального (головного) сервера.
3. Програмна «реалізація сервера» – модуль, який забезпечує авторизацію, автентифікаціюа також розмежування прав доступу користувачів різних рівнів та рангів до файлів.
4. База даних – серверна версія БД, у якій знаходяться відомості про користувачів, їхні файли та ін.

5. Файли – документи користувачів, до яких їм надано право доступу.

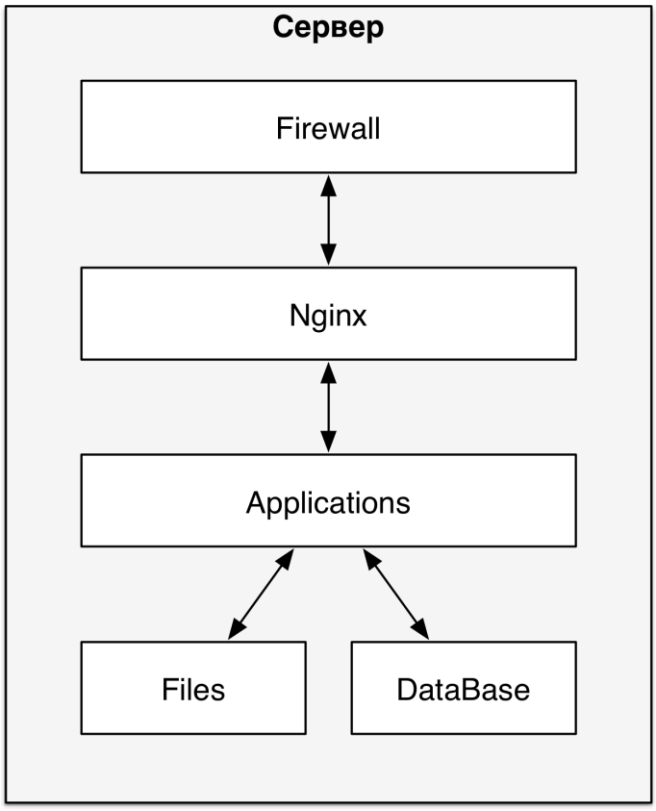


Рис. 2.4. Модель центрального сервера.

Основною проблемою такої моделі є актуальність даних, та потреба в клонуванні серверів з однаковими конфігураціями.

Застосовується й інший підхід для реалізації доступу до дата-центрів: «мережа доставки і дистрибуції контенту (Content Delivery Network, або Content Distribution Network, CDN) – географічно розподілена мережна інфраструктура, що дає змогу оптимізувати доставку та дистрибуцію контенту кінцевим користувачам в мережі Інтернет» [1]. Використання провайдерами CDN сприяє зростанню швидкості завантаження користувачами цифрових даних. На швидкість завантаження найбільше впливає те, наскільки далеко користувач знаходиться від самого сервера, оскільки основні затримки при передачі даних виникають на маршрутизаторах і залежать від їх кількості на маршруті. Перерозподіл контенту між кількома серверами у CDN значно скорочує мережні маршрути обміну даними.

При такій моделі організації хмарного сервісу зберігання даних фактичний час передачі файлів чи повідомлень між двома користувачами визначається

сумою часу передачі від першого користувача до дата-центру і від дата-центру до другого користувача. Не має принципової різниці, де буде знаходитися дата-центр – біля першого користувача, біля другого, чи приблизно по середині. У першому випадку дані будуть швидко доставлені на головний сервер дата-центру, але потім довго будуть передаватися з нього другому користувачу (рис. 2.5).

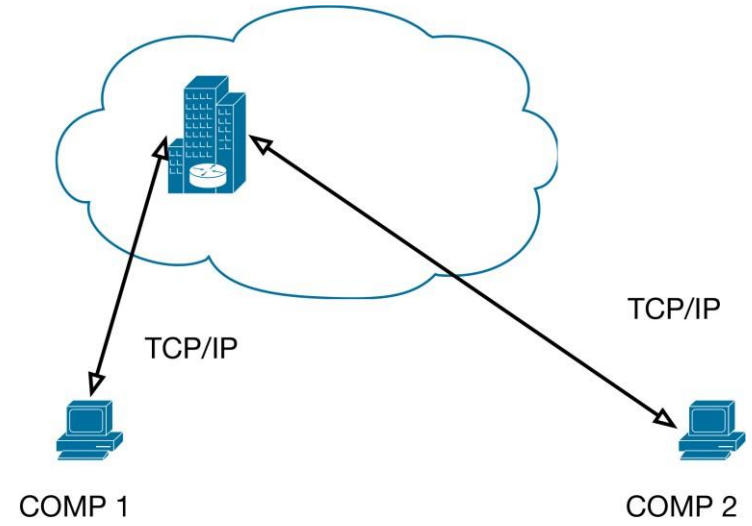


Рис. 2.5. Модель несиметричної передачі даних через дата-центр.

У протилежному випадку, дані будуть довше передаватися від першого користувача на головний сервер, але швидше з нього другому абоненту. Основним недоліком є низька швидкість передавання даних стандартним протоколом TCP/IP на значні відстані.

Якщо ж доповнити цю модель серверами-сателітами головного сервера у місцях локації користувачів (рис. 2.6), то отримаємо удосконалену модель доставки даних, завдяки встановленню постійних швидкісних маршрутів між ними, що навіть відеоконтент у форматі HD буде відносно стабільно передаватися, забезпечуватиметься швидке завантаження файлів більших розмірів, забезпечення відеотрансляцій в он-лайн режимі з високою якістю і низькими видатками на мережу.

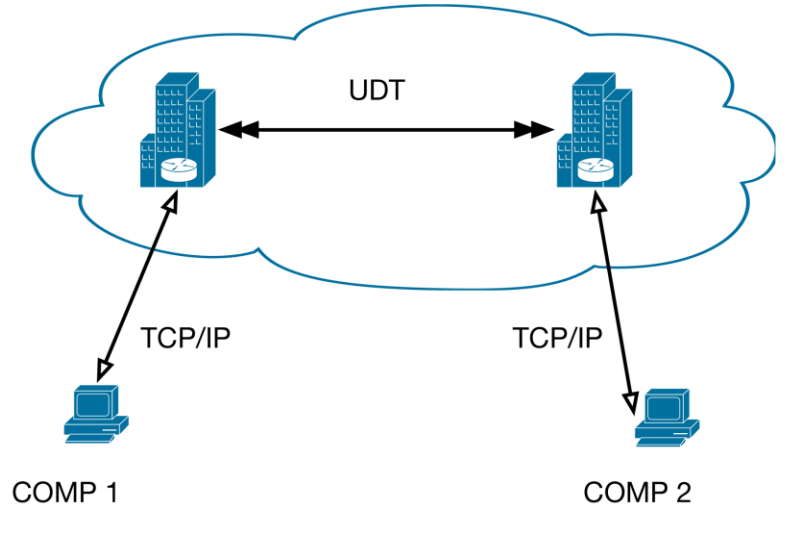


Рис. 2.6. Модель багатоточкового передавання даних через дата-центр.

Розміщення серверів-сателітів якомога ближче до користувачів збільшує пропускну здатність всієї системи. При такому підході дата-центр складається з географічно віддалених платформ, взаємодія яких дає змогу швидко опрацьовувати і задовільняти запити користувачів щодо завантаження чи вивантаження контенту.

Потужні дата-центри зазвичай складаються зі значного числа вузлів і розміщують свої сервери безпосередньо у локальних та регіональних мережах Інтернет-провайдерів.

Сервер- сателіт складається з (Рис. 2.7):

1. Firewall – захисний програмний модуль.
2. Nginx – веб-сервер для обробки запитів та переадресування їх на інтерфейсний модуль центрального (головного) сервера.
3. Application – Програмна реалізація сервера-сателіта – модуль, який визначає місце знаходження файлів, до яких бажає отримати доступ користувач та надає його.

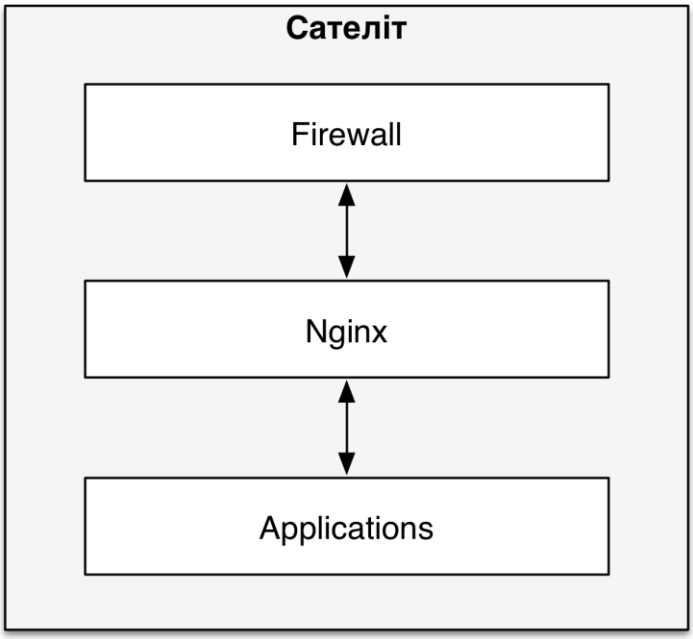


Рис. 2.7. Модель сервера-сателіта.

Таким чином, користувач розміщення та отримання своїх даних та файлів здійсює через сервер-сателіт, незалежно від того на якому головному сервері вони зберігаються.

2.2. Організація доступу до хмарного сховища.

Складність проєктування систем та мереж для передавання даних через хмарні сховища даних визначається складністю самих протоколів та методу з'єднання [22], які вони реалізують. Протоколи фактично є набором правил за якими взаємодіють системи та окремі елементи. Оскільки впровадження хмарних технологій відбувається вже на наявному апаратному забезпеченні, то для їх успішної реалізації застосовують програмні способи встановлення та налаштування протоколів.

Для забезпечення створення хмарних сховищ даних необхідно організувати:

- асинхронне передавання файлів декількома каналами зв'язку;
- потокове прочитання файлу і його передавання;
- прийом файла через декілька каналів зв'язку та його подальше кешування;

- потокове зчитування файла з кешу;
- синхронізацію завершення передавання файлу.

Відповідно до принципів функціонування Інтернет, запит може отримати лише один комп’ютер в мережі, тому для того, щоб запити на доступ до даних хмарного сховища, могли бути опрацьовані декількома реальними серверами на комп’ютері, що приймає запит, повинен бути запущений сервер-сателіт, який реалізує:

- отримання запиту від користувача;
- вибір реального сервера, який буде опрацьовувати запит;
- перенаправить запит користувача до реального сервера сховища даних;
- опрацювання відповіді сховища даних;
- переадресацію відповіді реального сервера на запит до користувача.

Таким чином, згідно описаного прийому, усі запити, що надходять у сховище, проходять через цей сервіс секрвера-сателіта. Оскільки, на опрацювання кожного запиту сателіту потрібен певний системний ресурс, то швидкодія головного сервера сховища даних обмежена як конфігурацією комп’ютерів з даними, так і конфігурацією того комп’ютера, на якому запущений сервер-сателіт.

Побудуємо модель передачі файла між двома користувачами (рис. 2.8):

1. Файл ділиться на частини-блоки однакового розміру.
2. Формується черга блоків.
3. Здійснюється передавання блоків через N каналів зв'язку (мультиплексування).
4. Здійснюється прийом переданих блоків файлу (демультиплексування).
5. Прийняті блоки формують чергу (пріоритет блоків визначається їх порядком у файлі).
6. Об'єднання блоків у файл і запис на диск приймаючого користувача.

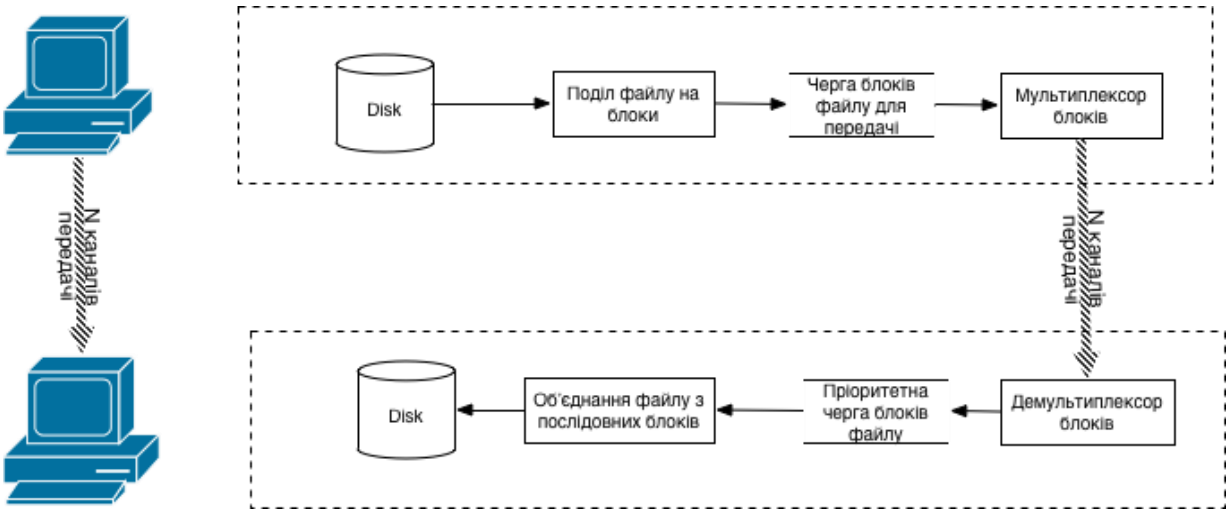


Рис. 2.8. Модель передачі файлів між двома користувачами.

Такий підхід можна вважати традиційним і він застосовується як класична модель передачі файлів у мережі. Дана модель підтримується існуючим апаратним та програмним забезпеченням комп'ютерних мереж. Проте в реальних розгалужених мережах для передавання файлів потрібно використовувати допооміжні вузли, сервери-сателіти, тощо. І чим таких проміжних вузлів буде більше на маршруті, тим складнішою буде і сама модель, і реальна мережа, яку можна буде втім, розкласти на декілька таких класичних моделей. Тобто у модель слід додати проміжні елементи, через які й представимо хмарні сховища даних, яких на маршруті може бути декілька (рис. 2.9). Це зумовлене специфікою використання протоколів класу TCP.

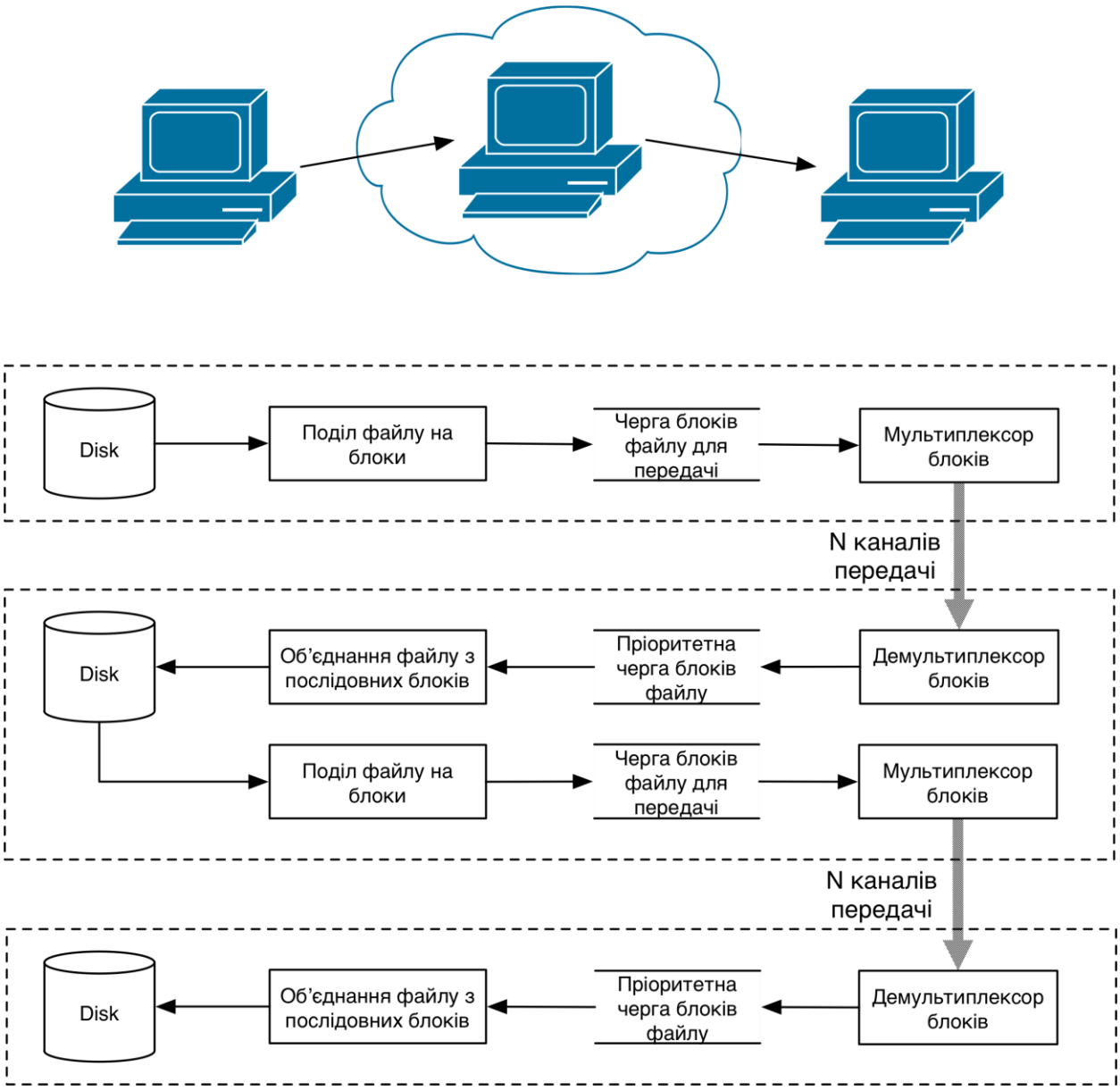


Рис. 2.9. Модель передачі файлів через проміжний елемент.

Це можна удосконалити, якщо в кожному проміжному елементі, не затримувати до закінчення прийняття усіх блоків файлу, а почати передавати їх ще під час надходження блоків, а кінцеве об'єднання файлів виконати аж на кінцевому вузлі (рис. 2.10). Для цього слід поміняти спосіб роботи проміжних елементів.

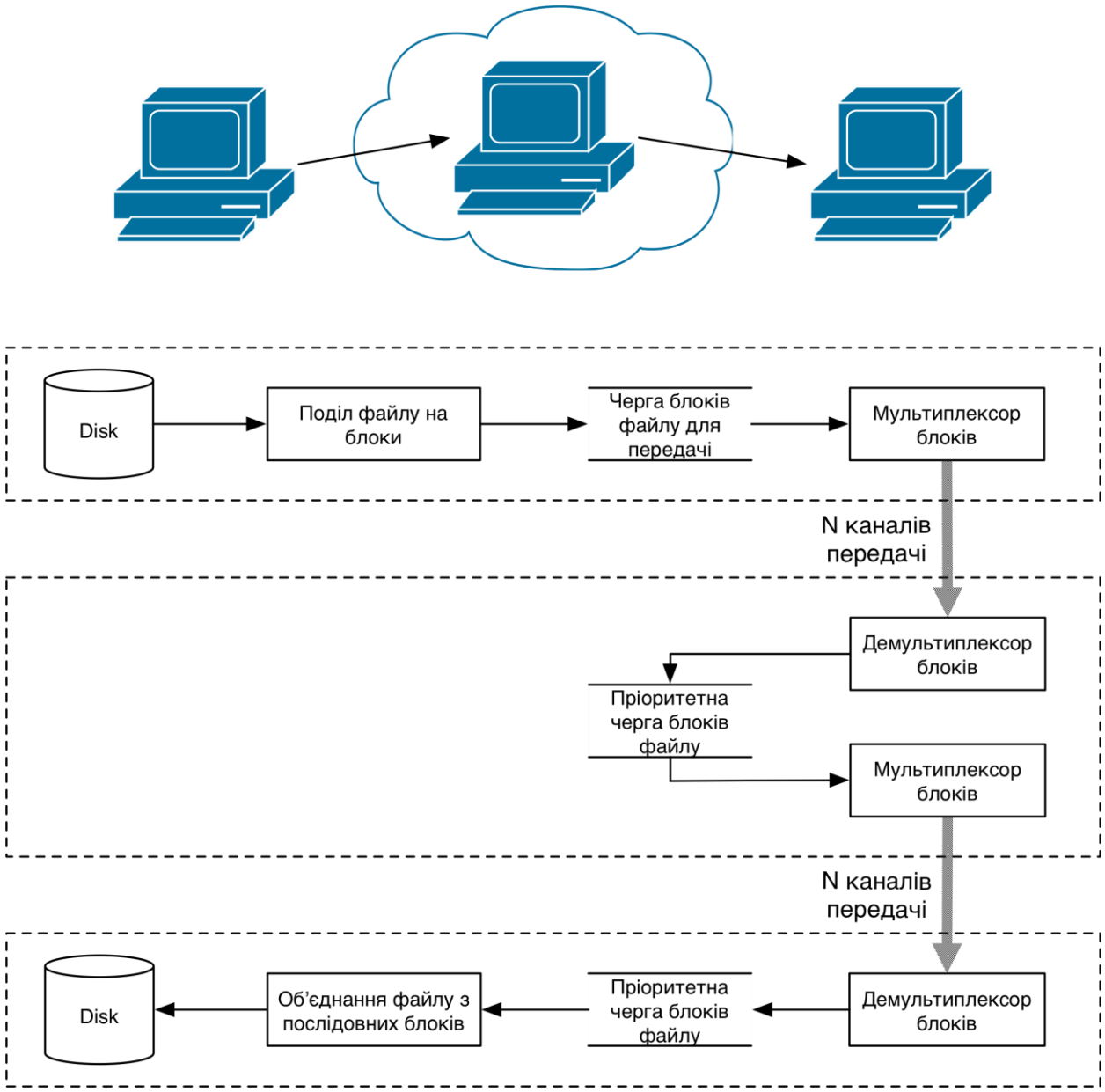


Рис. 2.10. Модифікована модель передачі файлів між двома користувачами.

Під час прийому блоків файлу у проміжному елементі слід проводити їх записування у чергу блоків за пріоритетами (пріоритет блоку визначається його порядковим номером при розбитті файлу на блоки). Окрім пріоритетної черги блоків на кожному проміжному елементі потрібно реалізувати лічильник переданих блоків, який відслідковував би номер кожного переданого блоку, а в пам'яті фіксував би останнє значення. І тоді, передачу блоків файлу можна розпочати, як тільки в голові черги з'явиться блок з номером 1. Після передачі кожного блоку лічильник збільшується на одиницю. Такий процес триватиме

доти, поки усі блоки файлу не будуть передані далі.

Втім, слід зазначити, що для кінцевих користувачів запропонована нами модель принципово не буде відрізнятися від старого підходу. Однак, завдяки такому підходу, ми отримаємо значний виграш у збільшенні швидкості передачі даних у вигляді файлів, адже зі своєї моделі передачі файлів ми виключили процеси збирання повного файлу на усіх проміжних елементах.

Затримка, яка неодмінно буде виникати у запропонованій моделі, зумовлена личе часом очікування наступного блоку файлу для подальшої передачі на кожному проміжному елементі, та варіантами передачі блоків різними каналами, кожен з яких матиме свої швидкісні характеристики.

2.3. Моделювання завантаженості хмарних сховищ даних.

Сучасні хмарні сховища даних не використовують свої потенційні функціональні можливості. Однією з причин цього є нестабільність мережного трафіку, як в середині самої хмари, так і зовні, що першочергово впливає на швидкість доступу до даних та й саму якість обслуговування.

Це в свою чергу потребує реалізації універсальної мережі з повністю розподіленою комутацією, де взаємодія між користувачами, пристроями і програмними додатками здійснювалася б через віртуальні з'єднання. Разом з тим, зростання числа споживачів послуг у хмарі підвищуються вимоги до мережного та серверного обладнання, транспортних протоколів, які повинні базуватися на дієвих математичних моделях оцінювання трафіку та мережних процесів [23, 25]. Так, зокрема, мережевий трафік визначається рядом чинників:

- поведінка користувачів,
- стабільність роботи серверного програмного забезпечення,
- протоколи передачі даних
- надійність використовуваного обладнання.

Таким чином, слід моделювати та досліджувати функціонування хмарного сервера через такі його параметри та характеристики, як швидкість і обсяг оперативної пам'яті, завантаженість центрального процесора, стан системних

процесів.

Модель повина показати завантаженість хмарного сховища, а потім її бажано перевірити на реальному прикладі, вимірюючи згадані параметри. З цією метою необхідно: виміряти трафік даних у хмарному сховищі, проаналізувати потік даних, встановити залежність рівня потоку даних від інших показників. Параметри сховища даних визначимо через вхідний /вихідний трафік, кількість запущених процесів, завантаженість процесорів, середнє навантаження на процесор, об'єм кеш-пам'яті.

$$L(CH(S_{cloud}), CH(S_{cloud})) \rightarrow Z \tag{2.1}$$

Завантаженість хмарного сховища даних представляють у вигляді функції:

$$CH(S_{cloud}) = \frac{IT \cdot OT \cdot LCPU \cdot MC}{V \cdot PN \cdot (FCPU + LCPU) \cdot ALCPU} \tag{2.2}$$

де, IT – вхідний трафік,

OT – вихідний трафік,

V – канал,

PN – кількість запущених процесів,

LCPU – завантаженість процесорів,

FCPU – простій процесорів,

ALCPU – середнє навантаження на процесор

MC – обсяг кеш-пам'яті.

Таким чином, на прикладі параметрів хмарного сховища даних було побудовано динамічну модель із урахування параметрів вхідного та вихідного трафіку, а також розподілу апаратних потужностей хмарного сервера. Для всіх процесів було використано їх схожість, що дало можливість застосувати цю модель для прогнозування та оцінки параметрів роботи серверів хмарних сховищ даних [24].

Класичний підхід до опису трафіку базується на припущенні, що вхідні потоки даних є сталими потоками Пуассонівського типу, тобто фактично являють собою суперпозицію нескінченного числа ординарних незалежних стаціонарних потоків малої інтенсивності. Проте дослідження реального трафіку доступу до хмарних сховищ даних, показують, що трафік з описаною вище

комутацією пакетів має зовсім іншу структуру, яка не може бути описана жодним із класичних загальноприйнятих підходів до побудови моделей такого типу. Це пов’язано з тим, що у ньому спостерігається деяка кількість досить сильних відхилень порівняно із низьким в середньому рівнем вигналу. Це значно погіршує і якість сигналу і його характеристики і не дає змоги описати його, користуючись згаданою вище моделлю.

Ще однією вихідною теорією для проектування хмарних систем доступу до даних була теорія масового обслуговування, яка досить добре описує процеси, що відбуваються в телефонних мережах, побудованих на основі комутації каналів.

Однак поява і повсюдне впровадження мереж із пакетною передачею даних, які практично витіснили системи з комутацією каналів, призвела ряду невирішених проблем:

- відсутність теоретичної бази функціонування системи пакетної передачі даних замість класичної теорії масового обслуговування;
- відсутність надійної методики розрахунку характеристик для кожного потоку, що відображав би інтенсивність надходження запитів на обслуговування до їх середнього числа;
- чітко визначених і визнаних характеристик якості систем функціонування хмарних сховищ даних;
- відсутність надійних засобів, що забезпечували б високу якість надання послуг доступу до хмарних сховищ даних.

Для розв’язання цих проблем необхідно вирішити наступні завдання:

– вивчити трафік даних у хмарних сховищах даних;

– здійснити аналіз потоків даних;

– дослідити залежність рівня фрактальності потоку даних від самоподібних характеристик окремих потоків, з яких він складається.

Вивчивши наявні класичні та новітні моделі, ми звернули увагу на модель трафіку як ланцюжка повідомлень [17]. В цій моделі вважається, що пакети даних передаються разом і можуть опрацьовуватися як одне ціле, а мережне

обладнання в кожному вузлі мережі прийматиме рішення про подальшу обробку ланцюжка повідомлень, що слідує за першим. Такий підхід усуває зайві операції аналізу кожного пакету. Однак, ця модель може бути застосована тільки для повідомлень, які мають одну кінцеву точку призначення. Стало зрозумілим, що практична реалізація існуючих транспортних протоколів і використовуване мережне обладнання для цієї моделі не можуть застосовуватись і причиною тому є такі чинники:

- поведінка користувача (його запити та звернення до хмарного сховища даних є нерегулярні);
- використання хмарного програмного забезпечення (потрібне стабільне високошвидкісне з’єднання);
- процеси генерування, структурування і пошуку потрібних даних;
- об'єднання трафіку;
- засоби адміністрування складників мережі, зокрема проміжних елементів;
- алгоритми оптимізації передавання та опрацювання пакетів даних;
- переструктурування мережі, зростання числа абонентів.

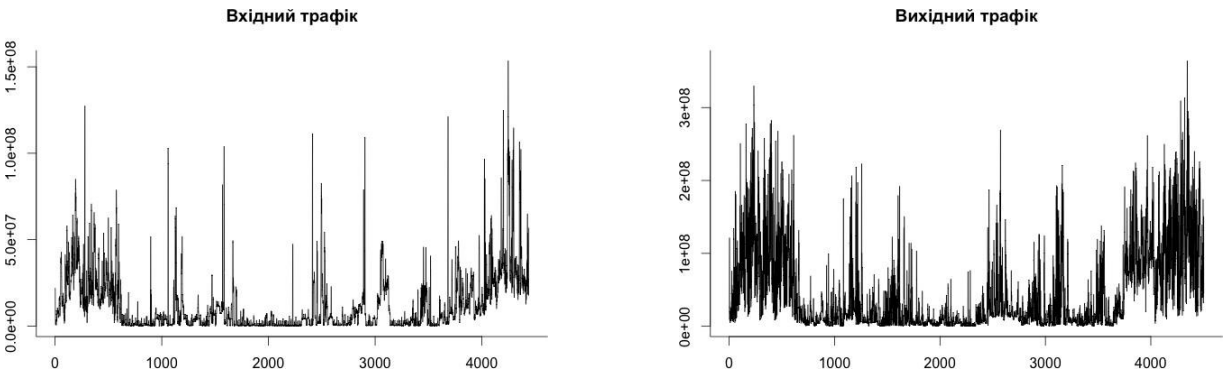
Для підтвердження наявності властивості самоподібності для різних потоків даних до та з хмарного сховища даних, проведено вимірювання вибраних характеристик мережного трафіку в умовах змінних характеристик сервера хмарного сховища даних [5]. Для експериментального дослідження використовувався сервер хмарного сховища даних, поділений на декілька віртуальних серверів, кожен з яких використовувався для оцінювання окремих характеристик.

Для моніторингу параметрів хмарного сховища даних застосовували утиліту типу клієнт- сервер Zabbix [29], яка дала змогу зібрати дані про стан мережі, мережні навантаження, а також характеристики сервера хмарного сховища даних в режимі реального часу, які входили в рівняння 2.2:

- вхідний/вихідний трафік;
- кількість запущених процесів;

- завантаженість і простій процесорів;
- середнє навантаження на процесор;
- об'єм кеш-пам'яті.

Ці дані збиралися упродовж тижня, тому можемо вважати, що вони відображають реальний стан використання хмарного сховища даних (рис. 2.11-2.13).



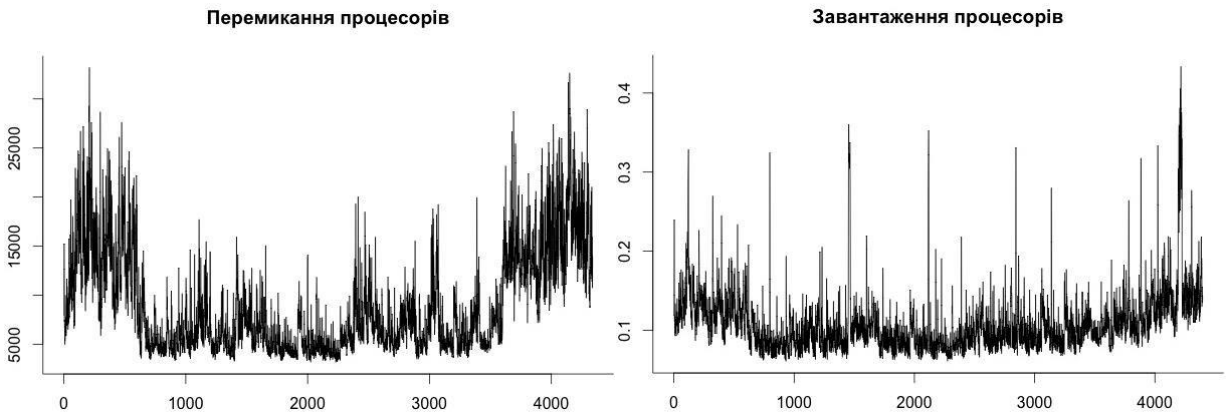
a) b)

Рис. 2.11. Залежності вхідного (а) і вихідного (б) трафіків.

Оскільки отримані дані аналізувалися упродовж тижня, то на графіках чітко проглядається добову періодичність.

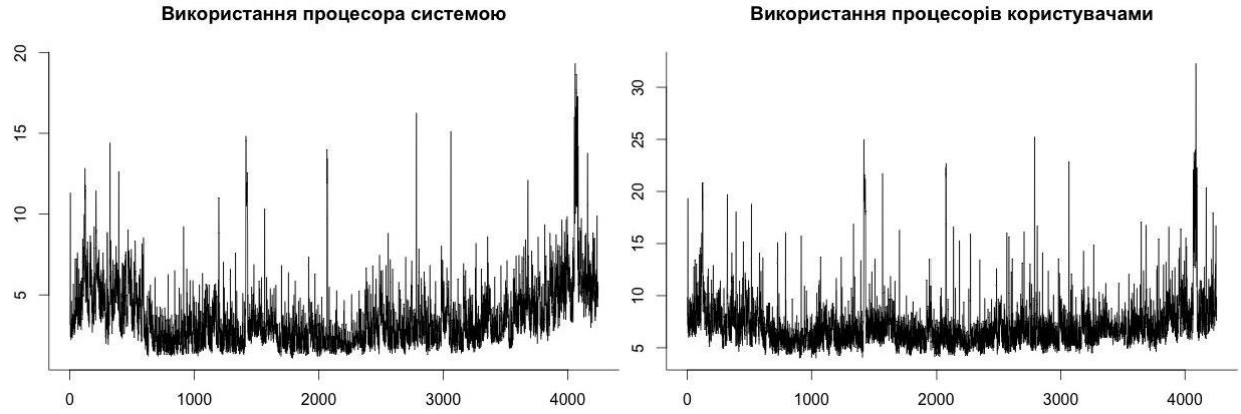
На рис. 2.12 показано ступінь завантаження регістрів процесорів, а на рис. 2.13. продемонстровано завантаженість процесорів системою і користувачами.

Подібно до графіку залежності вхідного і вихідного трафіку (рис. 2.11), ці залежності нагадують ту ж динаміку. Це пояснюється тим, що хмарне сховище даних забезпечує переважно введення, зберігання, опрацювання та виведення даних користувачів. Таким чином, основне навантаження на хмарні сховища даних здійснює вхідний та вихідний трафіки. Тому при моделюванні роботи хмарних сховищ даних можемо обмежитися моделями трафіку та завантаженості процесора.



a) b)

Рис. 2.12. Часові залежності перемикання (а) і завантаженості (б) процесорів.



a) b)

Рис. 2.13. Часові залежності використання процесорів системою (а) і користувачами (б).

Для наочної демонстрації здобутих залежностей побудуємо графічне представлення коефіцієнта кореляції (рис. 2.14). Видно, що точки на рисунку в групуються навколо прямої, кутовий коефіцієнт якої може бути визначений шляхом лінійної регресії. Визначивши кутовий коефіцієнт, отримали значення -0,21, для якого параметр Херста виявився рівним 0,89.

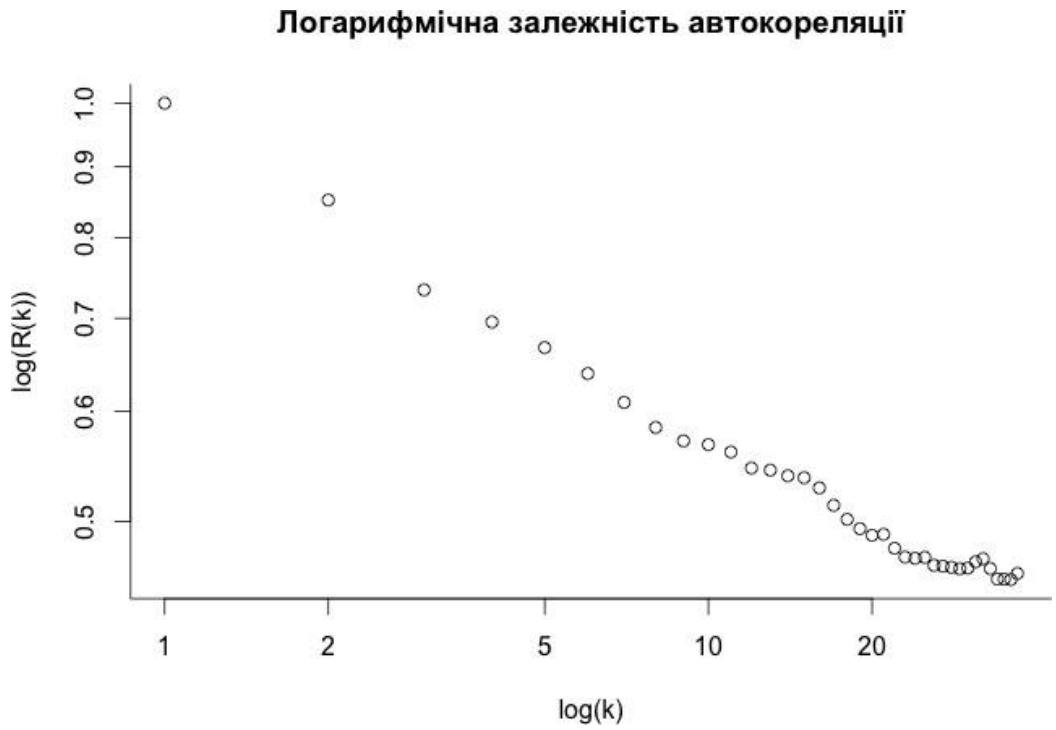


Рис. 2.14. Логарифмічна залежність коефіцієнту кореляції вхідного трафіку.

Для аналізу вихідного трафіку також побудовано графік коефіцієнта кореляції (рис. 2.15). Спостерігаємо подібну картину, як і для вхідного трафіку: точки групуються навколо прямої, кутовий коефіцієнт якої виявився рівним -0,2, а параметр Херста 0,9.

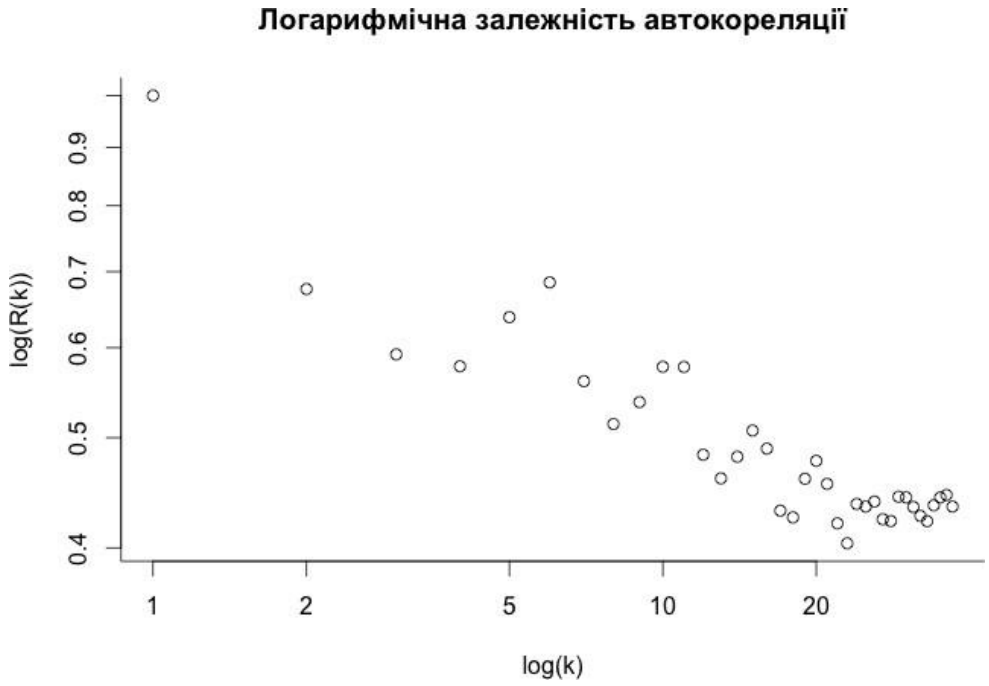


Рис. 2.15. Логарифмічна залежність коефіцієнту кореляції вхідного трафіку.

В обох випадках значення параметру Херста виявилось більшим 0,5 (0,5<H<1), що є достатньою передумовою для визнання процесу самоподібним.

При вивченні часової залежності трафіку було помічено закономірну сталу періодичність в ньому, що зумовило таке значення параметра Херста.

Таким чином, у результаті дослідження побудовано модель завантаженості хмарного сховища даних на основі аналізу вхідного і вихідного трафіку та завантаженості процесор на серверах хмарного сховища даних. Статистично значуще значення параметру Херста вказує на можливість моделювання і прогнозування завантаженості хмарного сховища даних на основі такого підходу.

Висновки до розділу 2

Як результат даного розділу відзначимо, що нами:

1. Розроблено модель хмарного сховища даних та охарактеризовано методи передачі даних.
2. Обгрунтовано дієздатність моделі мережного трафіку.
3. Здійснено моделювання функціонування та оцінювання завантаженості хмарного сховища даних.
4. Побудовано наочні зображення протоколів передачі даних від користувача до хмарного сховища даних і в зворотньому напрямку.
5. Побудовано модель хмарного сховища зберігання даних.

РОЗДІЛ 3.
МОДЕЛЮВАННЯ ТА АПРОБАЦІЯ РОЗПОДІЛЕНОЇ МЕРЕЖНОЇ АРХІТЕКТУРИ ХМАРНОГО СХОВИЩА ДАНИХ

3.1. Проектування архітектури системи.

Як було показано в попередньому розділі, специфіка функціонуваня хмарних сховищ даних полягає у використанні так званих розподілених систем. Проєктування такої розподіленої мережної архітектури хмарного сховища даних вимагає аналізу вимог до неї та її внутрішніх процесів.

Алгоритмічна складова системи повинна забезпечувати реалізацію необхідних функцій і володіти засобами організації процесів опрацювання, передавання та зберігання даних. З цією метою вона повинно бути:

- універсальною;
- функціональною;
- надійною;
- адаптивною;
- придатною до модернізації та масштабування;
- мати інтуїтивно зрозумілий і зручний інтерфейс;
- захищеним від зовнішніх впливів;
- здійснювати автоматичне документування усіх дій користувачів.

Програмна складова повинна розроблятися із застосуванням принципів структурного і модульного програмування, щоб відокремити кожну із задач системи і тим самим ще більше прискорити її функціонування.

Архітектура системи повинна відповідати таким вимогам:

- повинна реалізовувати горизонтальне масштабування серверів для збільшення продуктивності системи (як пристроїв для зберігання даних, так і серверів-сателітів для покращення доступу до даних);
- повинна мати можливість дублювання всіх важливих вузлів мережі та зберігання даних, для забезпечення відмовостійкості системи.

З цією метою до складу системи повинні входити:

1. Сховище даних.
2. Сателіт
3. Адаптивний DNS сервер.

Основними вимогами щодо надійності системи взято такі:

- система повинна функціонувати цілодобово, у неперервному режимі.
- потрібно здійснювати регулярне резервне копіювання даних, а резервні копії потрібно зберігати у різних місцях.
- захистити дані від руйнування незалежно від збоїв у системі.
- забезпечувати паралельне функціонування підсистем, щоб робота чи перебої однієї не впливали на функціонування інших.
- збої у самих даних не повинніпризводити до збою в роботі підсистем і навпаки.
- некоректні дії користувачів не мають спричиняти аварійну ситуацію.
- мінімізувати технічні помилки.
- могти працювати на різних операційних системах,
- використовувати типові СУБД (система управління базами даних).
- використовувати типових протоколів та інтерфейси взаємодії.
- відповідати вимогам міжнародних стандартів.

Розроблена система хмарного сховища даних має виконувати такі функції:

- реєстрація користувачів (реєстрація організації в системі; створення облікового запису адміністратора від організації);
- адміністративна частина (реєстрація користувачів організації; редагування їх профілів);
- забезпечення доступу до даних за стандартними протоколами (HTTP; HTTPS; FTP; FTPS; SFTP);
- опрацювання можливих помилок (сховища; сателітів; сценарію; можливих перешкод чи розірвання зв’язку; надмірне перевантаження);
- фіксацію дій користувачів у журналах.

Проаналізувавши описаний вище функціонал, побудовано графічну модель системи (рис. 3.1).

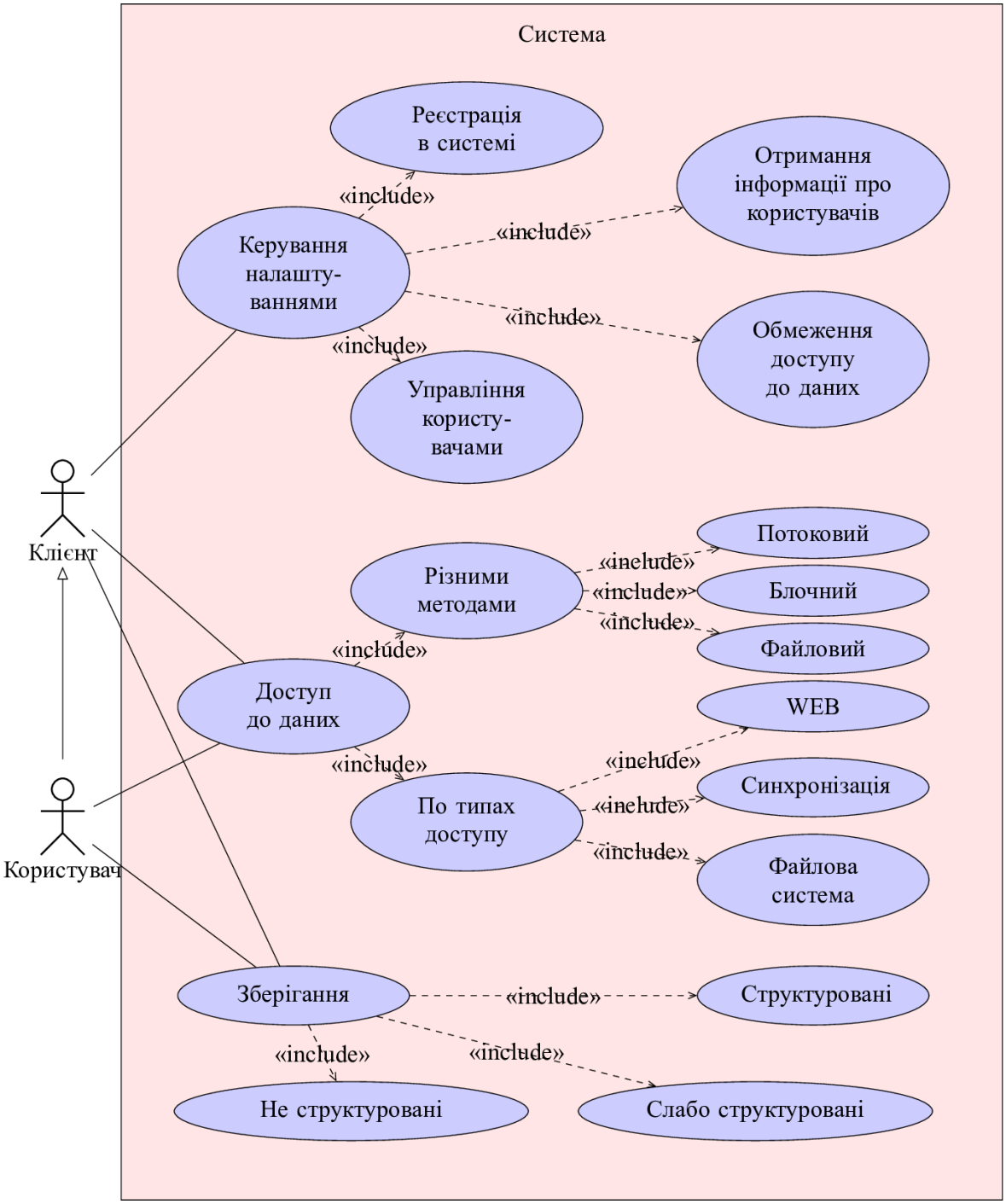


Рис. 3.1. Варіанти використання системи з точки зору Користувача та Клієнта.

У розділі 2 було встановлено, що базова архітектура системи складається із DNS сервера, сховищ даних (DC) та серверів-сателітів, що відображено на рис. 3.2).

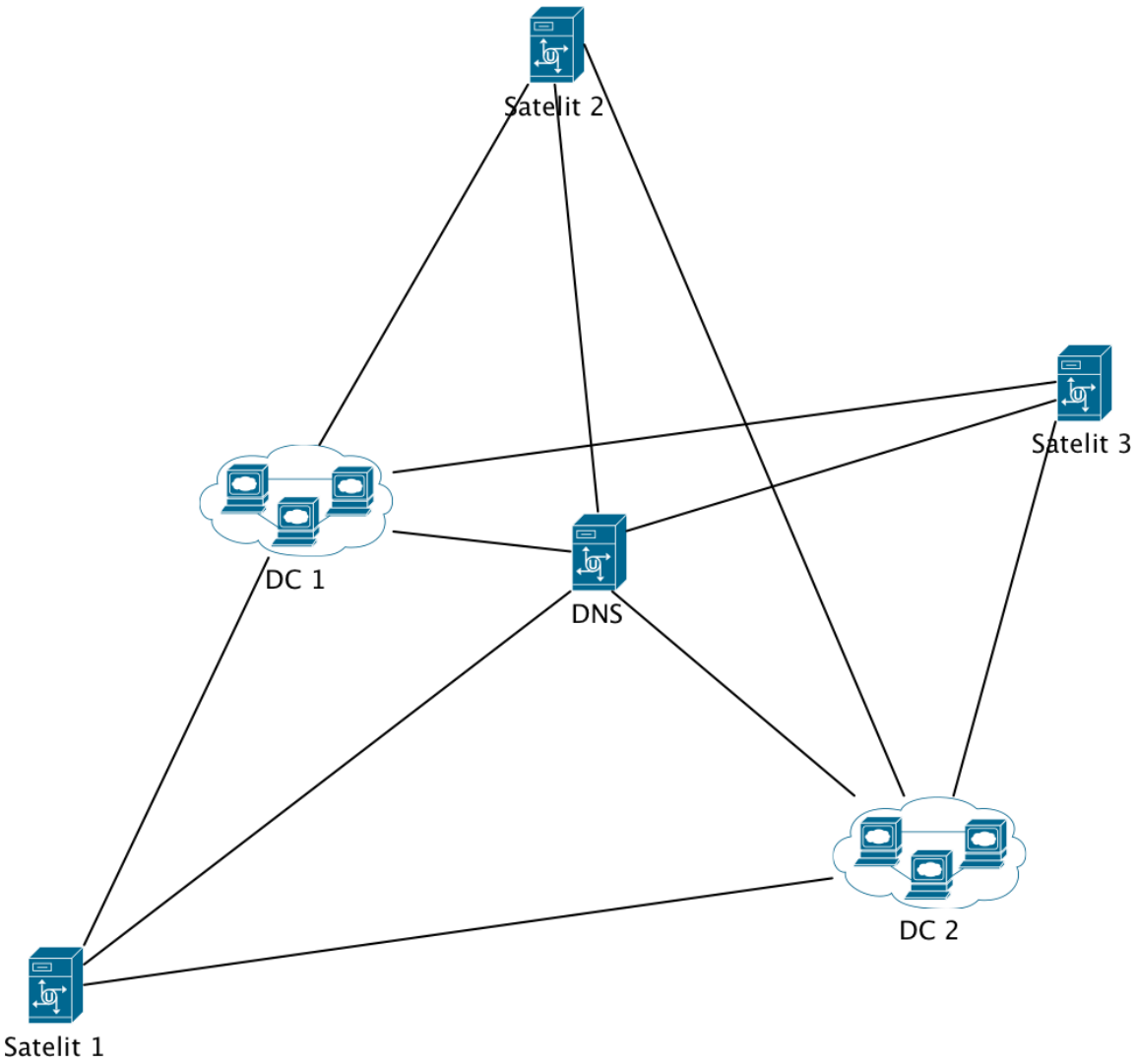


Рис. 3.2. Загальна архітектура системи хмарного сховища даних.

Два сховища мають підключатися як єдине сховище. Для цього на програмному рівні має бути забезпечена синхронізація даних між ними. Кожне зі сховищ повинно в свою чергу містити (рис. 4.4):

- два брандмауера;
- двах сервери з програмними додатками;
- два сховища з дисками;

що забезпечуватиме відмовостійкість системи у випадку втрати одного фізичного вузла мережі.

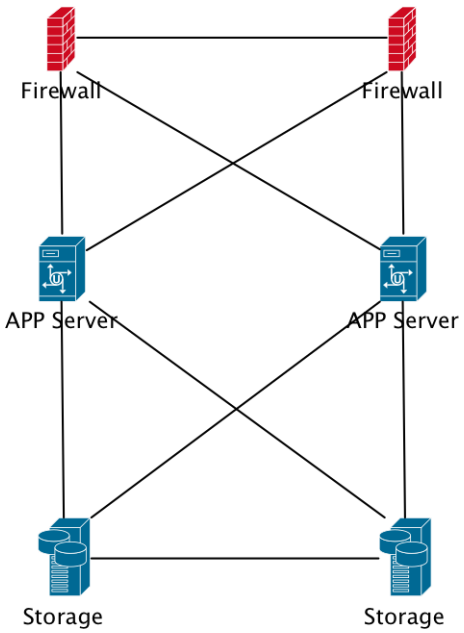


Рис. 3.3. Загальна архітектура одиничного сховища.

3.2. Дослідження та аналіз потоків даних в системі.

У роботі хмарного сховища даних, мають бути забезпечені як раціональна організація потоків з даними, так і підвищення їх передачі й опрацювання. Для реалізації цих завдань при проектуванні хмарного сховища даних слід здійснити аналіз інформаційних потоків, що дасть загальне уявлення про функціонування системи, сприятиме відображенню їх структури та динаміки, адже це є основою функціонування хмарного сховища даних.

Основними елементами хмарного сховища даних є (рис. 3.4):

- Сховище даних (рис. 3.5);
- Сателіт доступу до даних (рис. 3.6);
- DNS-сервер.

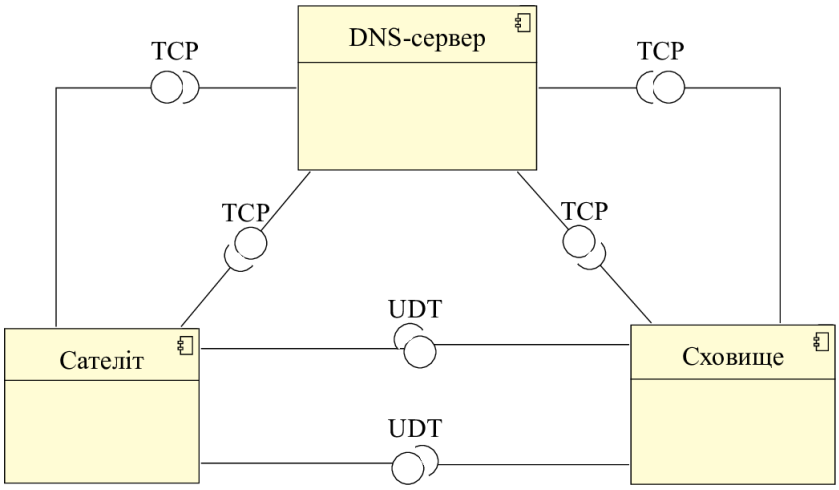


Рис. 3.4. Модель взаємодії основних елементів ХСД.

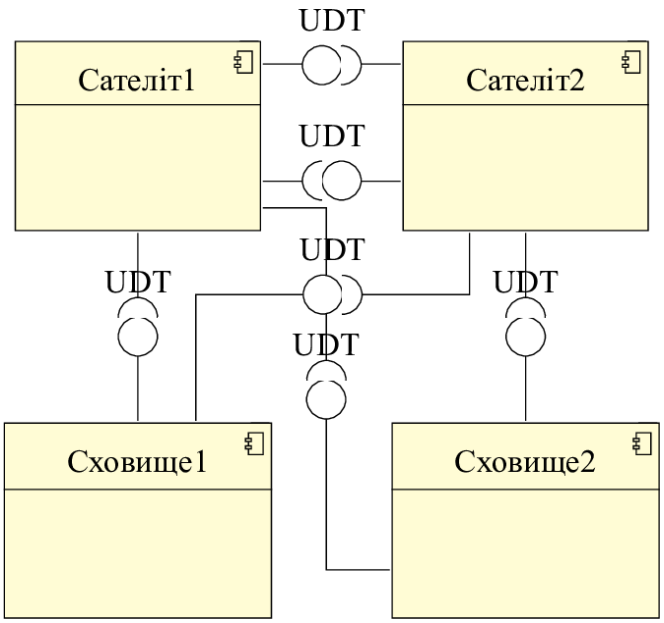


Рис. 3.5. Схема розміщення сателіту.

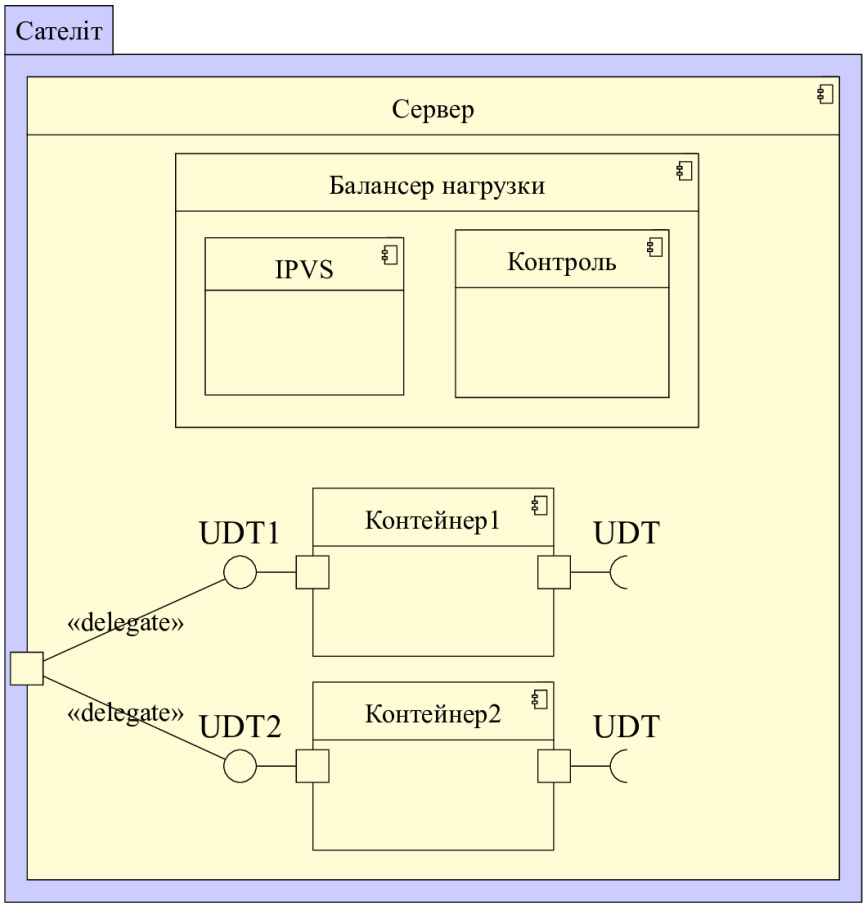


Рис. 3.6. Схема розміщення Сховища.

У свою чергу, елементи ХСД обмінюються даними за допомогою

протоколу UDT (рис. 3.7).

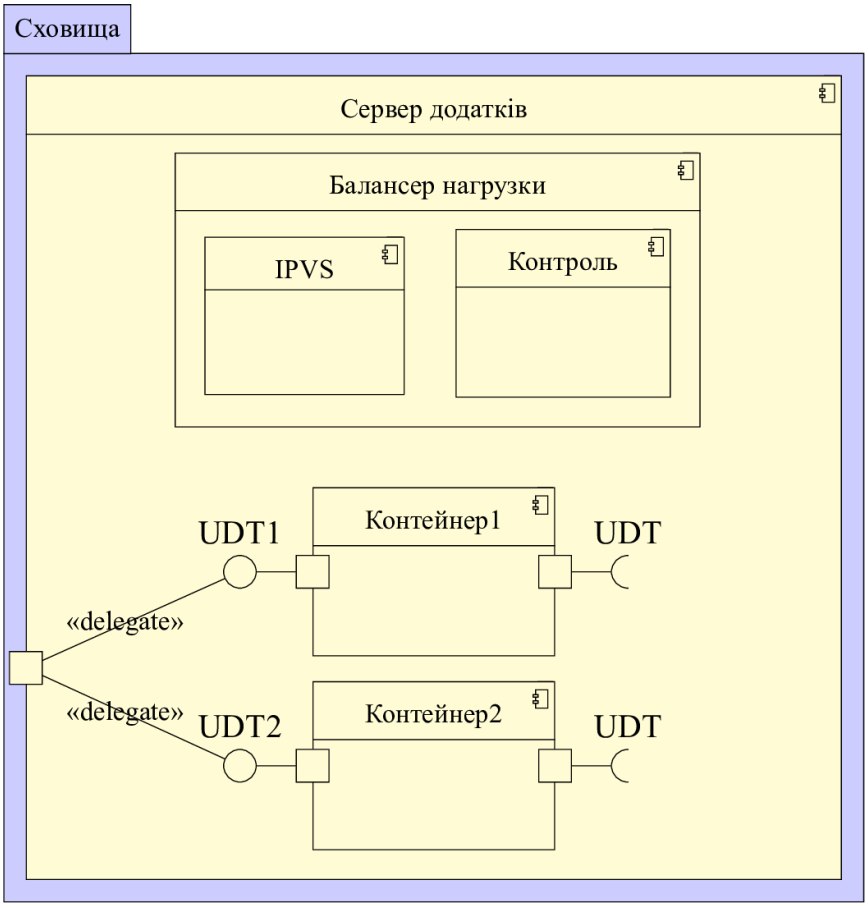


Рис. 3.7. Модель взаємодії сховищ та сателітів.

Для максимально ефективного використання ресурсів системи, кожний фізичний сервер поділяться на незалежні блоки (контейнери). Контейнери – це система віртуалізації на рівні операційної системи для запуску декількох ізольованих інсталяцій операційної системи. Для забезпечення функціонування системи із декількома контейнерами, як цілісної системи використано два основні їх типи:

- Контейнер «Балансер навантаження»;
- Контейнер програмного додатку.

«Балансер навантаження» виконує функції проксі-сервера, який відзначає доступність контейнерів та переадреадресовує їм отримані запити. Контейнер програмного додатку – це повноцінна копія програмного додатку який обробляє запит користувача та повертає йому результат.

За допомогою протоколу UDT контейнер сховища даних (рис. 3.8) отримує та

виконує запити, для чого в контейнері запущені взаємонезалежні процеси UDT-сервер та UDT-клієнт.

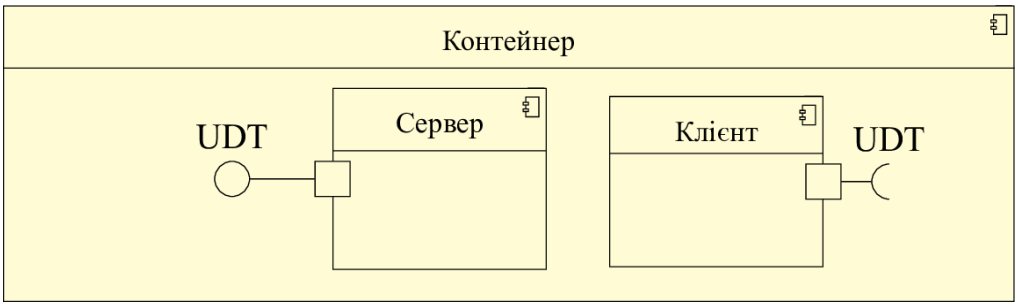


Рис. 3.8. Модель Контейнера сховища.

Контейнер сателіту (рис. 3.9) містить:

- UDT-сервер
- UDT-клієнт
- веб-сервер
- файловий сервер.

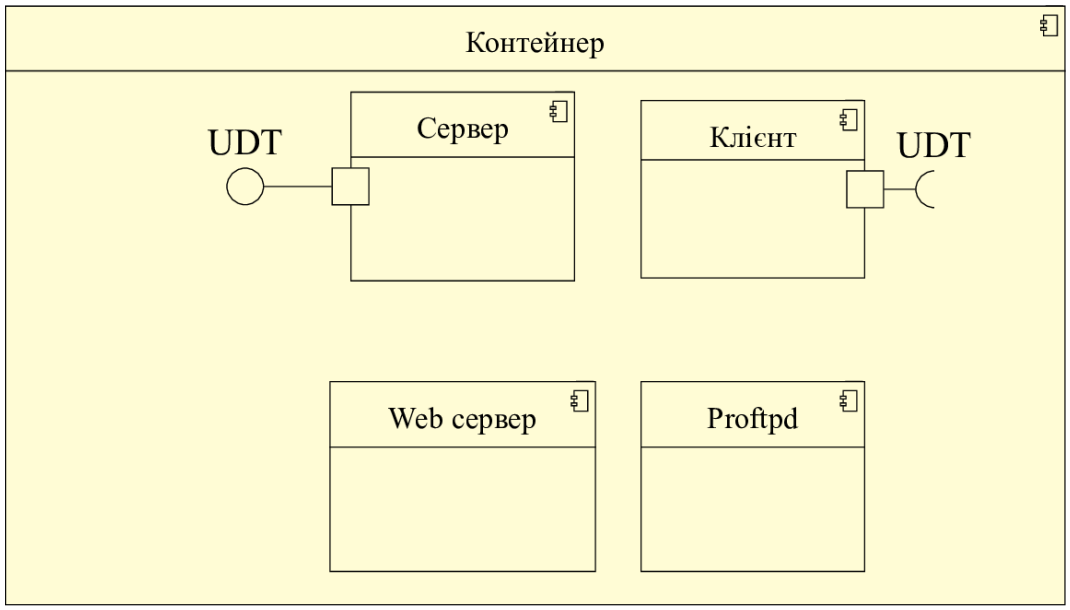


Рис. 3.9. Модель Контейнера сателіту.

Розклали функціонал системи на такі основні процеси:

1. Пошук найближчого сателіта.
2. Завантаження даних у сховище.

3. Доступ до даних.
4. Реплікація даних.

Пошук найближчого сателіта доступу до даних зображено на рис. 3.10.

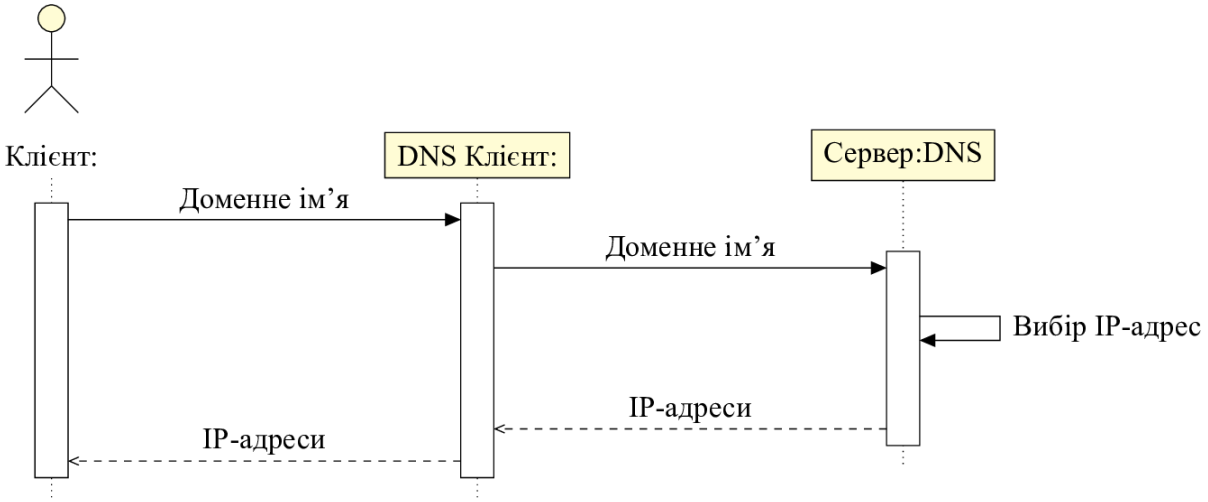


Рис. 3.10. Схема пошуку найближчого сателіта.

Завантаження даних у хмарне сховище реалізовано так, як показано на рис. 3.11.

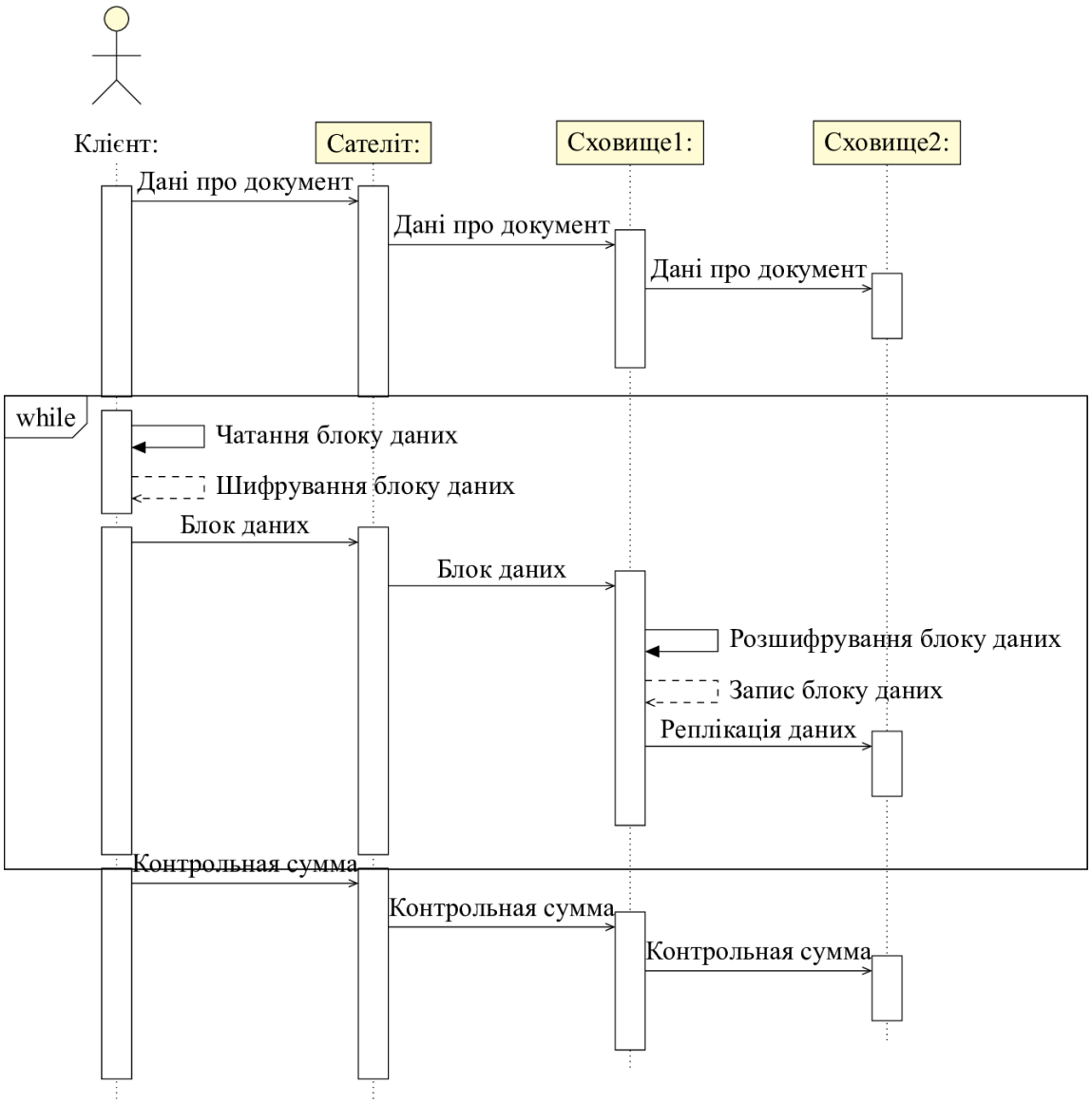


Рис. 3.11. Схема послідовності завантаження даних.

Реалізація доступу до даних показана на рис. 3.12.

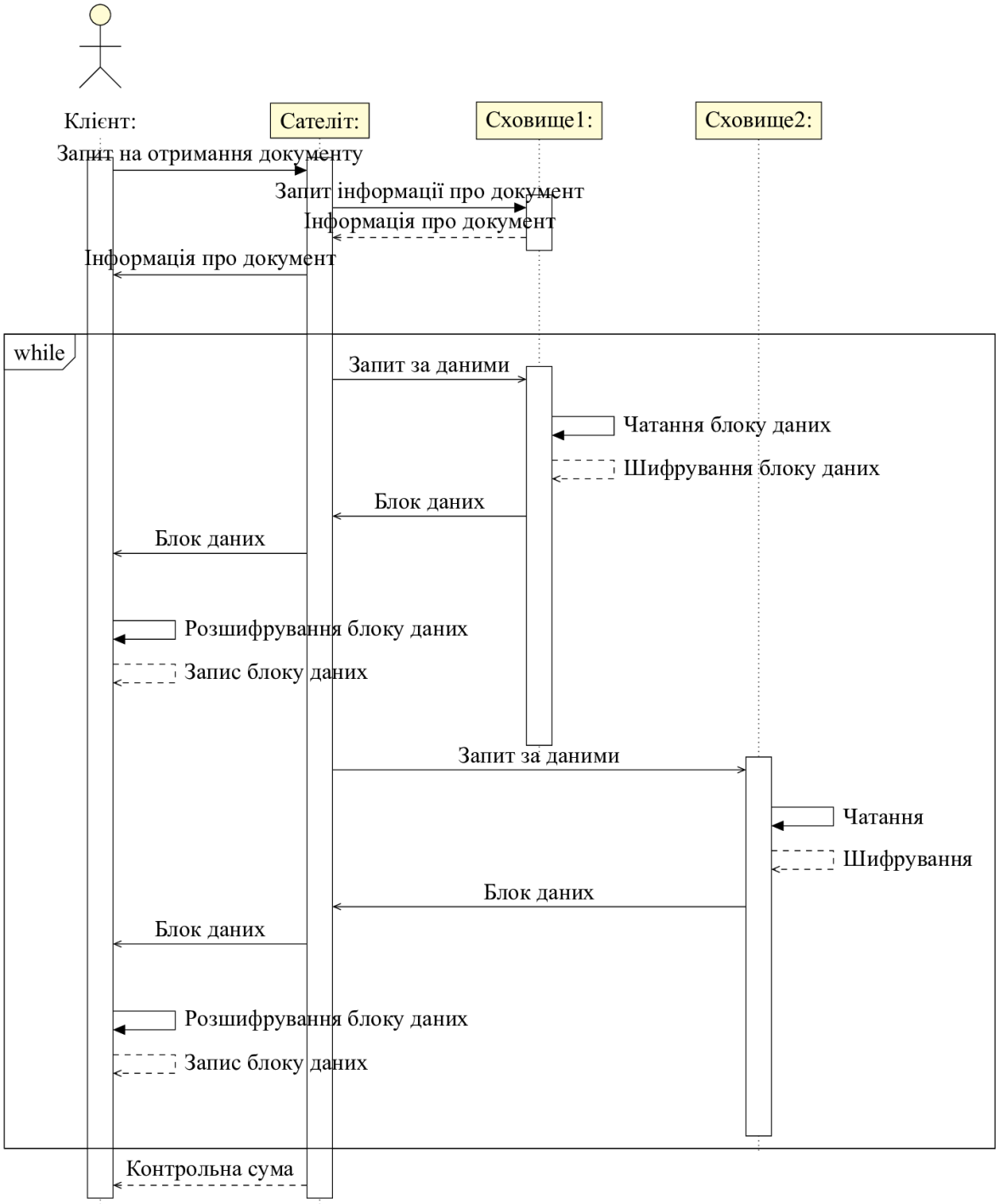


Рис. 3.11. Діаграма послідовності доступу до даних, розроблено самостійно.

3.3. Практична реалізація протоколів обміну даними в розподіленому хмарному сховищі даних.

Розроблена архітектура хмарного сховища даних забезпечує одночасну передачу і прийом даних, тому нами обрано асинхронний підхід до проєктування системи. Функціонування системи передачі даних подано на рис. 3.13, як сукупність

станів та переходів між ними.

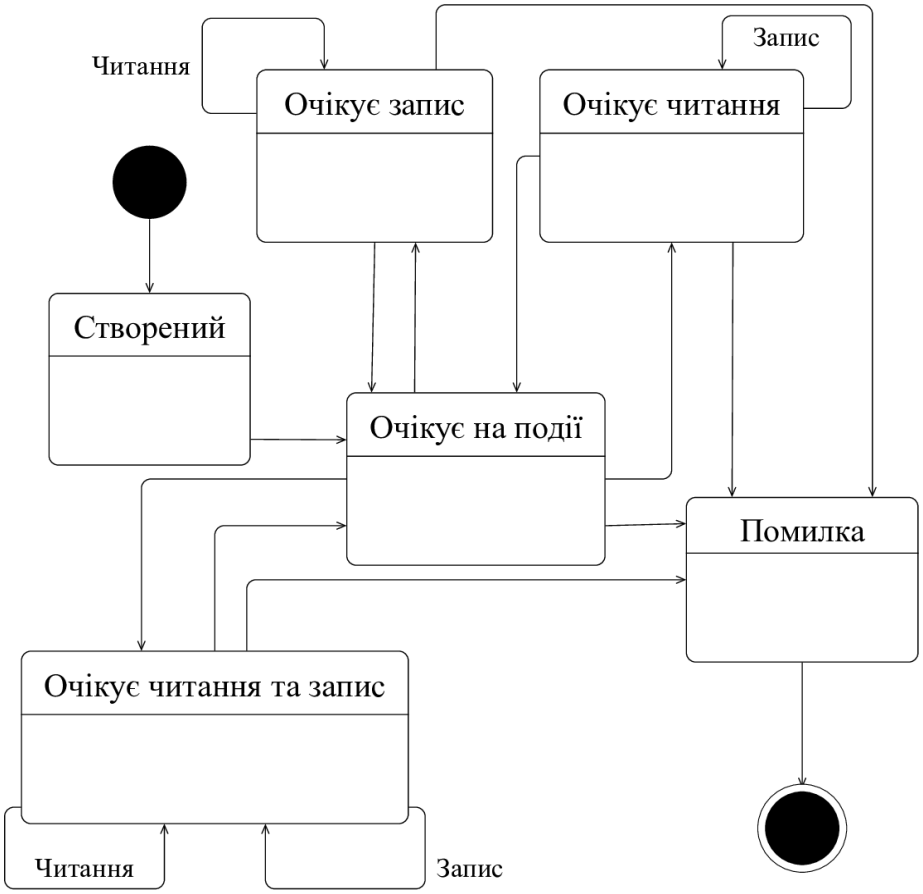


Рис. 3.13. Система передачі даних.

Після запуску процес переходить у статус «Очікування на події»:

- поява з'єднання (аутентифікація користувача; додавання нового з'єднання до загального пулу).
- нові вхідні дані (отримання, розшифрування та виконання передбачених дій із отриманим пакетом даних.
- перевірка передачі даних (наявність з'єднання з користувачем, формування пакетів даних, їх шифрування та передавання каналами зв'язку).
- помилка з'єднання (система оцінює помилку та приймає запрограмоване, згідно коду помилки, рішення.

Оскільки дані користувача зберігаються у хмарному сховищі даних у вигляді файлів, то передавання здійснюється у вигляді окремих пакетів даних (рис. 3.14).

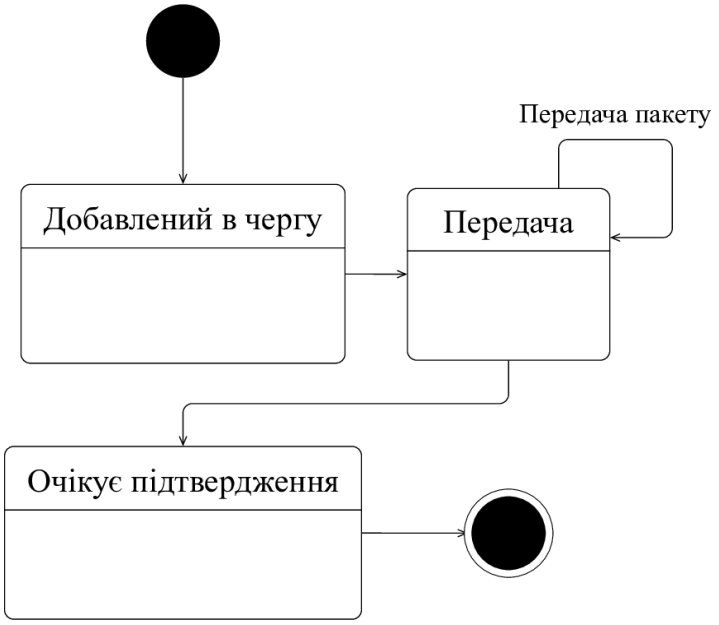


Рис. 3.14. Модель станів файлу для передавання.

Під час передавання даних система формує окремі пакети та додає їх у чергу для передачі даних. Після цього здійснюється перевірка на наявність стабільного з'єднання з кінцевим пунктом призначення для даних. Даля запускається шифрування та передача пакетів даних (з черги) по виділеному каналу зв'яку.

Після відправлення усього пакету даних, система позначає пакет, як «відправлений», а після повного отримання користувачем усього пакету даних, у систему відправляється пакет з підтвердженням, після чого пакет видаляється з черги передачі даних. У випадку розірвання з'єднання, система змінює статус відправлених пакетів даних, наново, для повної повторної передачі (рис. 3.15).

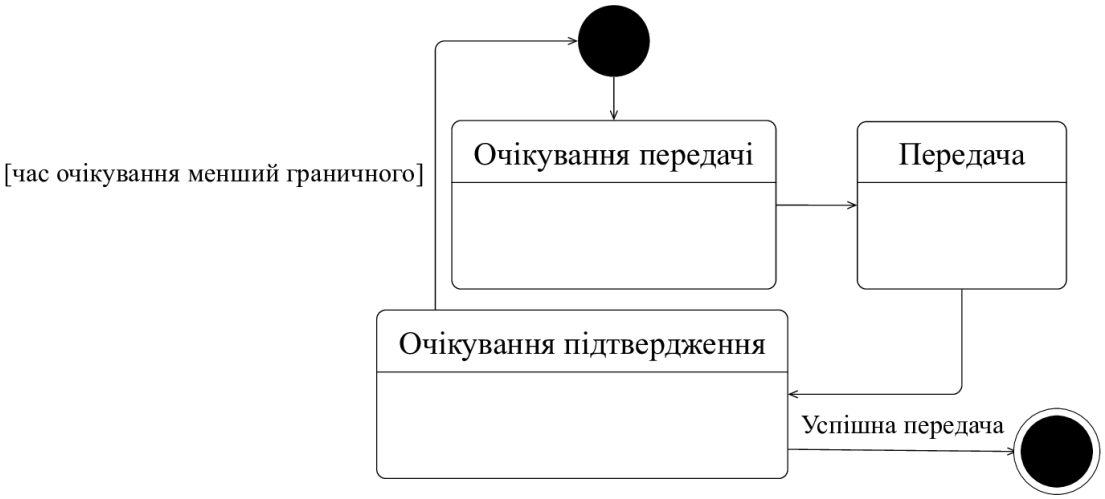


Рис. 3.15. Стани пакету передачі даних.

Модель функціонування системи передачі даних в хмарному сховищі даних подано на рис. 3.16.

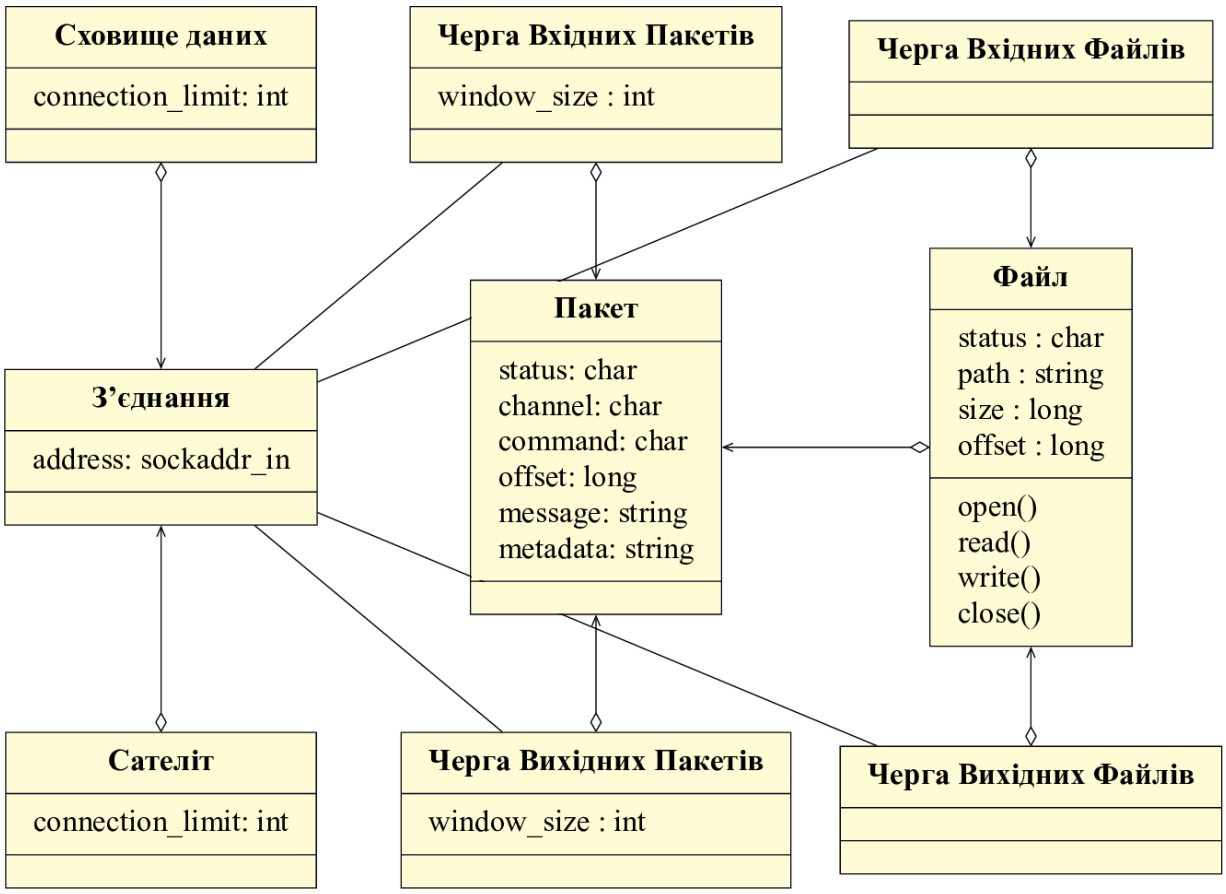


Рис. 3.16. Модель функціонування системи передачі даних в ХСД.

Як виявилось, модель передачі даних і в Сховищі даних, і в Сателіті практично подібні, що значно спростило побудову хмарних сховищ даних.

3.4. Порівняння характеристик хмарних сховищ даних.

Розглянемо переваги та недоліки використання найбільш популярних хмарних сховищ даних.

До *переваг впровадження ХСД* відносять:

- доступність (з будь-якого місця та в будь-який час за наявності під'єднання до Інтернет);

- низька вартість (оплачується тільки за те, що фактично використовується, або й узагалі користування певним обсягом дискового простору хмарного сховища даних може надаватися безкоштовно);
- значна економія пам’яті на жорсткому диску комп’ютера;
- спільний доступ до даних визначеному колу користувачів;
- гарантія цілісності даних забезпечується надавачем послуги.

До *недоліків застосування ХСД можна віднести*:

- зберігання конфіденційних даних;
- продуктивність з даними в “хмарі” може бути нижчою, ніж при роботі з локальними версіями;
- наявність стабільного та швидкісного під’єднання до мережі Інтернет.

При зберіганні даних у ХСД до них можливо отримати доступ з будь-якого пристрою, під’єднаного до Інтернет. Всі редагування, копіювання, сортування або вилучення даних автоматично відображаються на всіх пристроях одночасно. Вагомою перевагою зберігання даних на ХСД є можливість надавати спільний доступ до файлів та папок.

На сьогоднішній день користувачам пропонується багато хмарних сховищ даних. Найпопулярнішими хмарними сховищами даних є *Google Drive*, *One Drive* та *Dropbox*.

GoogleDrive – дає змогу користувачам зберігати свої дані на серверах компанії Google та надавати доступ іншим користувачами як для перегляду, так і для редагування в програмних додатках. У сервісі можна зберігати як документи, так і фотографії, музику, відео. Кожному користувачеві GoogleDrive надається безкоштовно 15 Gb дискового простору. Якщо ж цього недостатньо, то можна придбати додатково до 30 ТБ. Крім доступу до сервісу через веб-інтерфейс, є можливість доступу через клієнти для Windows, Mac OS, Android, iOS.

OneDrive – (до 2014 року Microsoft SkyDrive) інтернет-сервіс для зберігання та обміну файлами, створений за принципами хмарної організації. Він інтегрований

з Office 365, тому безпосередньо з програми можна створювати, редагувати та зберігати файли Excel, OneNote, PowerPoint і Word. Сервіс OneDrive дає змогу зберігати безкоштовно до 5 Гбайт даних. Передбачений попередній перегляд зображень у вигляді ескізів та слайдів.

Dropbox – хмарне сховище даних, що дає змогу користувачам зберігати свої дані на серверах в хмарі і надавати доступ іншими користувачам.

Додаток Dropbox можна встановити як на персональний комп'ютер, так і на мобільний пристрій. Однією з переваг Dropbox є легкість і інтуїтивність у використанні - потрібно завантажити файли в папку Dropbox, розшарити її, якщо хочете, або синхронізувати з потрібним пристроєм.

На відміну від згаданих вище сервісів ХСД, при роботі з Dropbox відредаговані файли не копіюються повністю на сервер – здійснюється передача тільки зміненої частини, попередньо стиснутої.

Наведемо їх порівняльну характеристику (табл. 3.1).

Таблиця 3.1.

Порівняльна характеристика найпопулярніших хмарних сховищ даних

Характеристика	***Dropbox***	***Google Drive***	***One Drive***
Безкоштовний обсяг дискового простору, Гб	2	15	15
Максимальний розмір файлу, Гб	10	5	10
Спільний доступ до даних	+	+	+
Тарифний план за 1 місяць	50 Гб – 10 $ 100 Гб – 20 $	25 Гб – 2,5 $ 100 Гб – 5 $	100 Гб – 20 $ 200 Гб – 40 $

Термін зберігання даних	Необмежений	Необмежений	Необмежений
Пряме посилання на завантаження	Так	Так	Ні
Можливість редагування документів MS Office	Так	Так	Так
Підтримка різних операційних систем	Windows, Mac OS, Linux	Windows, Mac OS	Windows, Mac OS
Наявність мобільної версії	Android, iOS, Windows Phone, Symbian, Bada	Android, iOS	Android, iOS, Windows Phone

Обираючи хмарний сервіс для зберігання даних слід враховувати потреби користувача, підтримку операційної системи, додатковий функціонал. Для того, щоб обрати найбільш підходящий сервіс, варто обирати експериментальним шляхом. Останнім часом хмарні сховища даних набули великої популярності і стали частиною нашого повсякденного життя, а хмарні технології інтенсивно розвиваються і надалі будуть ще більше використовуватися у повсякденному житті.

Висновки до розділу 3.

У даному розділі:

1. Спроєктовано архітектуру хмарного сховища даних.
2. Відповідно до спроєктованої архітектури проаналізовано та досліджено ефективність функціонування хмарного сховища даних.
3. Проведено практичну реалізацію хмарного сховища даних та досліджено процеси обміну даними через нього.
4. Проведено порівняльний аналіз хмарних сховищ даних.

ВИСНОВКИ

У магістерській роботі запропоновано варіант розв’язання актуального науково-практичного завдання розроблення моделі та методів забезпечення високої пропускної спроможності розподілених комунікаційних систем та хмарних сховищ даних, побудованих на їх основі. Основні результати магістерського дослідження полягають у тому, що:

1. Здійснено аналіз технологічних та функціональних проблем функціонування хмарних сховищ даних.

2. Удосконалено модель побудови хмарного сховища даних шляхом подання її як системи, яка побудована на основі новітніх протокольних засобів.

3. Побудовано модель завантаженості хмарних сховищ даних та перевірено її дієздатність на прикладі реального сховища.

4. Запропоновано удосконалений підхід перерозподілу навантаження між декількома джерелами даних, що дало можливість оптимізувати їх швидкодію.

5. Експериментально перевірено програмне забезпечення для передавання даних між вузлами хмарного сховища даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Патент US Patent 13/493,859. Content delivery network. Gagliardi, J.D., Munger T.S., Ploesser D.W. 2012. №5.

2. Патент US Patent 5,768,271. Virtual private network. Lespagnol A., Seid H.A. 1998. №6.

3. Струбицький Р. П., Струбицький П. Р., Шаховська Н. Б. Аналіз загроз хмарковим сховищам даних та методів їх захисту. *Наукові праці. Комп'ютерні технології*. 2013. №217. С. 35–38.

4. Струбицький Р. П., Шаховська Н. Б. Аналіз підходів до моделювання хмарних сховищ даних. *Актуальні проблеми економіки* : Науковий економічний журнал. 2013. №11. С. 263–269. 13.

5. Струбицький Р. П. Самоподібна модель завантаженості хмарних сховищ даних. *Вісник Національного університету «Львівська політехніка».* 2015. № 814. С. 147–156.

6. Таненбаум Э. С. Компьютерные сети. СПб: Издательский дом «Питер», 2012. 955 с.

7. 24 Huo Y., Wang H., Hu L., Yang H.. A cloud storage architecture model for data-intensive applications *Computer and Management (CAMAN), 2011 International Conference on. IEEE*. 2011. С. 1–4.

8. Al-Fares M., Loukissas A., Vahdat A.. Al-Fares M. A scalable, commodity data center network architecture. *ACM SIGCOMM Computer Communication Review*. 2008. №38. С. 63–74.

9. Avelar V. Guidelines for Specifying Data Center Criticality. APC. 2007. Режим доступу до ресурсу: http://www.lamdahellix.com/assets/contents/files/122_whitepaper.pdf.

10. Foster I., Zhao Y., Raicu I., Lu S. Cloud computing and grid computing 360-degree compared. *Grid Computing Environments Workshop*. 2008. С. 1–10.

11. Dikaiakos M. D., Katsaros D., Mehra P. Cloud computing: distributed internet computing for IT and scientific research. *Internet Computing, IEEE*. 2009. №13. С. 10–13.

12. Dekeyser S. Watson R. Extending google docs to collaborate on research papers. *University of Southern Queensland*, Australia. 2008. №23. С. 1–11. №37

13. Fall K. R., Stevens R.W. TCP/IP Illustrated, Volume 1: The Protocols. 2011.

850 с.

14. Feng W., Tinnakornsrisuphap P. The failure of TCP in high-performance computational grids. *Supercomputing, ACM/IEEE 2000 Conference*. IEEE. 2000. С. 37–37.

15. Herrick D. R. Google this!: using Google apps for collaboration and productivity *Proceedings of the 37th annual ACM SIGUCCS fall conference*. 2009. С. 55–64.

16. Hewitt C. ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing. *IEEE internet computing*. 2008. №5. С. 96–99.

17. Jain R., Routhier S. Packet Trains-Measurements and a New Model for Computer Network Traffic. *IEEE Journal on Selected Areas in Communications*. 2006. №6. С. 986–995.

18. Jones M. T. Anatomy of a cloud storage infrastructure. *IBM developer works* (November 30, 2010). 2010. Режим доступу до ресурсу: http://www.ibm.com/developerworks/cloud/library/cl-cloudstorage/cl- cloudstorage-pdf.pdf.

19. Katabi D., Handley M., Rohrs C. Congestion control for high bandwidth-delay product networks *ACM SIGCOMM Computer Communication Review*. 2002. №4. С. 89–102.

20. Miller L. J. The ISO Reference Model of Open Systems Interconnection: A first tutorial. *Proceedings of the ACM '81 conference*. 1981. С. 283– 288.

21. O'Reilly T. What is web 2.0., 2009. – 12 с.

22. Russell T. Telecommunications Protocols / Travis Russell., 2000. – 427 с.

23. Stallings W. High-speed Networks and Internets: Performance and Quality of Service. 2002. 715 с.

24. Strubytsky R. Organization of Cloud Storage Data in Distributed Systems *Proceeding of the XIIIth International Conference "Modern problems of radio engineering, telecommunications and computer science* (tcset'2016)" Lviv–Slavske, Ukraine, February 23–26, 2016. С. 463–467.

25. Thajchayapong S., Peha J. M. Mobility patterns in microcellular wireless networks. *IEEE Wireless Communications and Networking Conference. Proceedings*. 2003. №3. С. 52–63.

26. Thompson K., Miller G. J., Wilder R.. Wide-area Internet traffic patterns and characteristics *IEEE Network*. 1997. №11. С. 10–23.

27. TransLight: a global-scale LambdaGrid for e-science [T. DeFanti, C. De Laat, J. Mambretti та ін.]. *Communications of the ACM*. 2003. №11. С. 34–41.

28. Transport protocols for high performance: Whither TCP / [A. Chien, T. Faber, A. Falk та ін.]. *Communications of the ACM*. 2003. №46. С. 42–49.

29. Vacche A. D., Lee S. K. Mastering Zabbix, 2013. 358 с.

30. Vascellaro J. E., Morrison S. Google gears down for tougher times. *Wall Street Journal*. 2008. С. A1.