МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.
«____»_____ 2019 р.

МАГІСТЕРСЬКА РОБОТА

НА ТЕМУ:

Аналіз протоколів доступу в хмарних середовищах

Освітньо-кваліфікаційний рівень "Магістр"
Спеціальність 123 – "Комп'ютерна інженерія"

Науковий керівник роботи: _____ (підпис) Л.О. Шумова (ініціали, прізвище)

Консультант з охорони праці: _____ (підпис) Я.О. Критська (ініціали, прізвище)

Студент: _____ (підпис) Є.О Луганський (ініціали, прізвище)

Група: КІ-17дм

Сєвєродонецьк 2019
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень магістр
Напрям підготовки
(шифр і назва)
Спеціальність 123 Комп'ютерна інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри
І.С. Скарга-Бандурова
«_____» _____20____р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Луганському Євгену Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Аналіз протоколів доступу в хмарних середовищах

керівник проекту (роботи) Шумова Лариса Олександрівна, к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "18" 10 2018 р. № 220/48
2. Строк подання студентом роботи 09.01.2019 р.
3. Вихідні дані до роботи дані, зібрані під час проходження науково-дослідної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Теоретико-методологічні аспекти 2. Аналіз систем віртуалізації і зв'язку між компонентами 3. Практична розробка додатку на основі методу єдиного критерію 4. Охорона праці та безпека в надзвичайних ситуаціях. Екологія

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Електронні плакати.

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

Охорона праці та безпека в надзвичайних ситуаціях. Екологія	Критська Я.О., ст.викладач		

7. Дата видачі завдання 18.10.2018 р.

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналітичний огляд літератури за темою роботи	1.09.18 – 10.10.18	
2	Аналіз систем віртуалізації і зв'язку між компонентами	11.10.18 – 24.10.18	
3	Аналіз методів оцінки продуктивності систем віртуалізації	25.10.18 – 07.11.18	
4	Практична розробка додатку на основі методу єдиного критерію	08.11.18 – 28.11.18	
5	Розгляд питань охорони праці та безпеки в надзвичайних ситуаціях	29.11.18 – 11.12.18	
6	Оформлення пояснювальної записки	12.12.18 – 20.12.18	
7	Оформлення презентації роботи	21.12.18 – 6.01.19	
8	Підготовка та подання магістерської роботи до захисту	7.01.18 – 16.01.19	

Студент _____ _____
(підпис) (прізвище та ініціали)

Науковий керівник _____ **Шумова Л.О.**
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Луганський Е.А. Аналіз протоколів доступу в хмарних середовищах.

Проаналізовано існуючі на даний момент актуальні протоколи доступу до середовища хмарних обчислень. Розглянуто параметри та характеристики, які безпосередньо впливають на підвищення їх ефективності. Створено універсальний тест продуктивності протоколів доступу, який дозволяє порівнювати споживання обчислювальних ресурсів хмарної інфраструктури. Розроблено метод отримання даних для єдиного критерію. Розроблено програмний вимірювальний інструмент, що дозволяє автоматизувати процес збору експериментальних даних.

Ключові слова: протоколи доступу, хмарні обчислення, єдиний критерій, тест, дані, програмне забезпечення.

АННОТАЦИЯ

Луганский Е.А. Анализ протоколов доступа в облачных средах.

Проанализированы существующие на данный момент актуальные протоколы доступа к среде облачных вычислений. Рассмотрены параметры и характеристики, которые непосредственно влияют на повышение их эффективности. Создан универсальный тест производительности протоколов доступа, который позволяет сравнивать потребление вычислительных ресурсов облачной инфраструктуры. Разработан метод получения данных для единого критерия. Разработан программный измерительный инструмент, что позволяет автоматизировать процесс сбора экспериментальных данных.

Ключевые слова: протоколы доступа, облачные вычисления, единый критерий, тест, данные, программное обеспечение.

ABSTRACT

Luhanskyi E.A. Analysis of access protocols in cloud environments.

Analyzed the currently existing topical protocols for access to the cloud-computing environment. The parameters and characteristics that directly affect the improvement of their effectiveness are considered. A universal test of the performance of access protocols was created, which allows comparing the consumption of computing resources of the cloud infrastructure. A method for obtaining data for a single criterion has been developed. A software-measuring tool has been developed that automates the process of collecting experimental data.

Key words: access protocols, cloud computing, uniform criterion, test, data, software.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ХО - Хмарні обчислення

ІТ - Інформаційні технології

ПЗ - Програмне забезпечення

ПК - Персональний комп'ютер

ВПС - Високопродуктивна система

ВРС - Віртуальний робочий стіл

ВМ - Віртуальна машина

ДКТ - Дискретна косинуса трансформація

ОЗП - оперативне запам'ятовуючий пристрій

ЦПП - Центральний процесорний пристрій

ШПД - Швидкість передачі даних

REST - Representational State Transfer (Передача репрезентативного стану)

API - Application programming interface (Інтерфейс програмування додатків)

FTP - File Transfer Protocol (Протокол передачі файлів)

HTTP - Hypertext Transfer Protocol /(Протокол передачі гіпертексту)

HTTPS - HTTP over Secure Sockets Layer (Рівень HTTP поверх Secure Sockets)

VDI - Virtual Desktop Infrastructure (інфраструктура віртуальних робочих столів)

RDP - Remote Desktop Protocol (прокол віддаленого робочого столу)

PCoIP - Personal Computer over Internet Protocol (протокол доступу до персонального комп'ютера через Інтернет)

TCP - Transmission Control Protocol (протокол контролю передачі)

UDP - User Datagram Protocol (протокол датаграм користувача)

ВСТУП

З початком інформаційної епохи стала зростати потреба суспільства обробляти все більші обсяги інформації, а також надавати доступ до даних в довільний момент часу. Для вирішення цього завдання використовується підхід, що забезпечує зберігання інформації з використанням середовищ хмарних обчислень (ХО). Ідея середовища ХО була вперше висунута в 1961 році Джоном Маккарті, проте в зв'язку з відносно низькою пропускною спроможністю каналів передачі даних на той момент, не могла бути реалізована.

Середа хмарних обчислень - це програмно-апаратна модель засобів обчислювальної техніки, що дозволяє отримувати віддалений доступ до обчислювальних ресурсів в будь-який момент часу. Середа ХО дозволяє динамічно виділяти необхідну програмного забезпечення (ПЗ) процесорний час і пам'ять в залежності від поточного навантаження на це ПЗ. При цьому користуються програмним забезпеченням, що виконується в середовищі ХО, забезпечується за допомогою мережі Інтернет.

У **стандарті NIST SP-800-145**, опублікованому Інститутом стандартів і технологій (NIST), хмарні обчислення розуміються як модель надання повсюдного мережевого доступу до загального пулу обчислювальних ресурсів, які динамічно надаються споживачеві хмари на вимогу, з огляду на його запити. Удосконалення програмних алгоритмів і все більш гостро стоять проблеми з ефективністю використання обладнання призвели до нового якісного витка розвитку в області інформаційних технологій - популяризації хмарних обчислень.

Міграція користувачів у віртуальне середовище неможлива без забезпечення комфортної роботи з наданими через неї сервісами (робота з офісними додатками, перегляд сторінок в браузері, робота з мультимедіа і т.д.). Якість сервісів, а, отже, і задоволеність користувачів, залежить як від продуктивності самої віртуальної інфраструктури, так і від продуктивності термінальних протоколів доступу до неї, за допомогою яких користувачі здійснюють доступ до віртуальних робочих столів. Для вимірювання продуктивності протоколів необхідно провести комбінований аналіз споживання основних значущих ресурсів віртуальної інфраструктури (споживання процесорних ресурсів, оперативної пам'яті і пропускної здатності мережі).

Проблема продуктивності VDI протоколів була відомою, тому цим питанням займалося багато дослідників серед яких M. Sridharan, W. Vereecken, LeThanhMan, A. Berryman, Calyam. У своїх роботах вони зачіпали проблематику оцінки продуктивності VDI протоколів, серед яких Microsoft RDP, VMware PCoIP, Citrix ICA, Hewlett-Packard RGS і віртуальних робочих столів в цілому. Для оцінки продуктивності дослідники вибирали критерії, на основі яких згодом визначався кращий з протоколів, а самі дослідження проводилися експериментально, під впливом різних базових задач. Недоліком даних досліджень можна вважати **ізольованість**

результатів, отриманих по кожному з критеріїв продуктивності і **неможливість** зв'язати і висловити їх вплив на загальну продуктивність один через одного.

Удосконалення програмних алгоритмів і все більш гостро стоять проблеми з ефективністю використання обладнання призвели до нового якісному витку розвитку в області інформаційних технологій - популяризації хмарних обчислень і появи систем віртуалізації. Всебічний розвиток і множинні плюси зробили серверну віртуалізацію дуже популярною серед великих і середніх компаній. Технологія віртуалізації користувальницьких місць отримує все більш широке поширення в першу чергу в сферах освіти, промисловості та охорони здоров'я.

Інфраструктура віртуальних робочих столів (DVI), все частіше впроваджується з метою знизити вартість і витрати обслуговування призначених для користувача додатків, а також збільшити їх масштабованість та керованість шляхом переходу з традиційних робочих станцій до віртуальних робочих столів.

У даній роботі пропонується використовувати підхід єдиного критерію, суть якого полягає у приведенні всіх характеристик (**швидкість передачі даних, формат повідомлення, перешкодозахищеність, стійкість до змін, безпеку**) протоколів до єдиного виміру і наочно оцінити внесок кожної з них в загальному споживанні ресурсів. Цей критерій дозволить розробити програмне забезпечення, завданням якого є **автоматичне тестування протоколів, з можливістю вибору тестових завдань різної складності і характеристик**.

Об'єкт дослідження: процеси, що забезпечують надійність доступу до середовища хмарних ресурсів.

Предмет дослідження: моделі і методи передачі даних.

Мета роботи: підвищення ефективності використання протоколів доступу до середовища хмарних ресурсів шляхом тестування і програмна реалізація вимірювального інструмента.

Вирішуються такі завдання для досягнення поставленої мети в магістерській роботі:

– проаналізувати існуючі на даний момент протоколи доступу до середовища хмарних ресурсів. Скласти єдиний критерій оцінки ефективності протоколів доступу до середовища хмарних ресурсів, в одному інтервалі часу.

– визначити тестове завдання і методику тестування. Порівняти час виконання основних етапів тестового завдання і аналіз навантаження на процесор, оперативну пам'ять і мережу. Зробити висновки за результатами тестування.

– розглянути параметри і характеристики, що безпосередньо впливають на підвищення ефективності протоколів доступу до середовища хмарних обчислень.

– розробити програмне забезпечення, завданням якого є автоматичне тестування протоколів, з можливістю вибору тестових завдань різної складності і характеристик.

РОЗДІЛ 1
ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ АСПЕКТИ

1.1 Історія розвитку хмарних технологій та їх види

Слово «хмара» (cloud) використовувалося в 1990-х роках для метафоричного позначення Інтернету: тоді Глобальна мережа представлялася чимось загадковим, невизначеним в своїх просторових межах, не відрізнятись від своїх внутрішніх елементів і швидко мінливих. Зафіксоване в статті під заголовком «ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing» визначення «хмарних обчислень» говорить: «Це той випадок, коли інформація постійно зберігається на серверах в Мережі і тимчасово зберігається на стороні клієнта - наприклад, на настільних комп'ютерах, планшетах, ноутбуках, міні-комп'ютерах і так далі »[1, с. 96-99].

Вперше ідею «хмарних обчислень» озвучив Д. Ликлайдер в 1960 році. Його ідея полягала в тому, що кожна людина на планеті буде підключений до мережі, з якої він буде отримувати не тільки дані, але і програми. Інший вчений, Джон Маккарті висловив ідею про те, що обчислювальні потужності будуть надаватися користувачам як послуга (сервіс)[2].

У 90-і рр. відбувається швидкий розвиток глобальної мережі - інтернет, що надає опосередкований вплив на розвиток хмарних технологій. Значно збільшилася пропускна здатність мереж, розширилася географія охоплення. Поряд з розвитком комп'ютерних мереж удосконалилися апаратні технології, з'явилися багатоядерні процесори, значно збільшився обсяг сховищ інформації.

Поява першої технології, близькою до сучасного розуміння терміна «cloud computing», приписується компанії Salesforce.com, заснованої в 1999 році. Дана компанія стала першою компанією, яка надала доступ до свого додатком через сайт, по суті, дана компанія стала першою компанією, яка надала своє програмне забезпечення за принципом - програмне забезпечення як сервіс (SaaS). Наступним кроком стала розробка хмарного веб-сервісу компанією Amazon в 2002 році. Даний сервіс дозволяв зберігати інформацію і робити обчислення.

У 2006 Amazon запустила сервіс під назвою Elastic Compute cloud (EC2) як веб-сервіс який дозволяв його користувачам запускати свої власні додатки. Наступним свою технологію поступово ввела Google, почавши з 2006 року пропозиція SaaS сервісів під назвою «Google Apps», а потім і моделі надання платформи як сервісу (PaaS) під назвою "Google App Engine". І, нарешті, свою пропозицію анонсувала компанія Microsoft, презентувавши її на конференції PDC в 2008 році під назвою «Azure Services Platform»

Дуглас Мініфі IT-директор великої компанії The Schumacher Group, яка займалася управлінням відділеннями невідкладної допомоги лікарень і організацією праці лікарів. Перед

ним постало питання: «Чим все-таки повинна в першу чергу займатися наша фірма - розробляти програмне забезпечення або використовувати його для управління медичними ресурсами?"

З цього питання і почалося в The Schumacher Group дослідження абсолютно нового IT-феномена під назвою "хмарні обчислення". Тим не менше, більшість IT-директорів продовжують покладатися на власні серверні інфраструктури з однієї простої причини: вони не впевнені, що хмарні обчислення вже готові для широкого виходу в світ. Причому, якщо вірити повідомленням в присвячених цій технології форумах, головне питання полягає зовсім не в тому, чи достатньо вона надійна для IT-середовищ.

Набагато більше IT-керівників турбують інші аспекти. Вони не впевнені в безпеці своїх даних, які виявляються в руках оператора "хмари". Вони вважають, що не зможуть ефективно управляти хмарними ресурсами. Вони підозрюють, що провайдери не розкривають всі деталі, що підтримує хмарну середу інфраструктури. Вони бачать в новій технології загрозу своїм обчислювальним центрам і навіть персоналу. Все це в підсумку стримує розвиток ринку хмарних обчислень.

Але що б там не говорили про хмарні обчислення, ясно одне: розвиток цієї технології просто неможливо ігнорувати. Варто відзначити, що ідея оренди додатків, платформ розробки, обчислювальних потужностей, сховищ і будь-яких інших "хмарних" сервісів повторює шлях Інтернету від експериментальної системи до серйозного призначеному для користувача інструменту. Технологія хмарних обчислень здатна в корені змінити вигляд інформаційних технологій.

Незважаючи на коливання серед IT-директорів, все більше постачальників хмарних сервісів активно просувають свої послуги в передчутті майбутнього прориву в цій галузі. Найзріліші пропозиції надходять сьогодні з боку Amazon, Google і Salesforce.com, які мало не щодня додають в свої сервіси все нові функції.

IBM, яка підключилася до досліджень Google в сфері хмарних обчислень, проводить агресивний маркетинг архітектури Blue Cloud, спеціально розробленої для даної технології. І деяких великомасштабні фірми, прагнучі не упустити шансу, укладають з Intel партнерські угоди по створенню великомасштабної тестової системи хмарних обчислень.

Деякі компанії вже зараз пропонують операторам зв'язку, кабельним компаніям і постачальникам послуг Інтернету багатий асортимент апаратних засобів для реалізації цієї технології.

Оскільки «хмари» - поняття збірне, має сенс класифікувати їх за будь-якою ознакою. Нижче наведені класифікації «хмар», одна з яких запропонована виданням InfoWorld, а інша - комерційні директором компанії Parallels, одного з лідерів ринку систем віртуалізації.
InfoWorld пропонують ділити все «хмари» на шість типів:

– sAAS - безпосередньо додатки у вигляді сервісу (наприклад, Zoho Office або Google Apps);

– службові обчислення - наприклад, віртуальні сервери;

– веб-сервіси в «хмари» - оптимізовані для роботи в віртуальному середовищі інтернет-сервіси (наприклад, системи інтернет-банкінгу);

– pAAS - «платформа як сервіс», тобто нове покоління веб-додатків, які дають можливість вибудовувати набір можливостей за бажанням користувача (наприклад. Live Mesh від Microsoft);

– mSP - провайдер керованих сервісів (Managed Service Provider), які обслуговують сервіс-провайдерів (наприклад, вбудовані антивірусні сканери для поштових порталів);

– комерційні Платформи для сервісів - об'єднання PaaS і MSP (наприклад, Cisco WebEx Connect).

Хмари поділяють на приватні, публічні, гібридні і кланові [3].

Приватне хмара, (англ. Private cloud) - інфраструктура, призначена для використання однією організацією, що включає кілька споживачів (наприклад, підрозділів однієї організації), можливо також клієнтами і підрядниками даної організації. Приватне хмара може перебувати у власності, управлінні та експлуатації як самої організації, так і третьої сторони (або будь-якої їх комбінації), і воно може фізично існувати як всередині, так і поза юрисдикцією власника.

Публічне хмара, (англ. Public cloud) - інфраструктура, призначена для вільного використання широкою публікою. Публічне хмара може перебувати у власності, управлінні та експлуатації комерційних, наукових і урядових організацій (або будь-якої їх комбінації).

Гібридне хмара, (англ. Hybrid cloud) - це комбінація з двох або більше різних хмарних інфраструктур (приватних, публічних або суспільних), що залишаються унікальними об'єктами, але пов'язаних між собою стандартизованими або приватними технологіями передачі даних і додатків (наприклад, короткочасне використання ресурсів публічних хмар для балансування навантаження між хмарами).

Хмара спільноти, (англ. Community cloud) - вид інфраструктури, призначений для використання конкретним співтовариством (кланом) споживачів з організацій, що мають спільні завдання (наприклад, місії, вимог безпеки, політики, і відповідності різним вимогам). Громадське хмара може перебувати в кооперативній (спільної) власності, управлінні та експлуатації однієї або більше з організацій спільноти або третьої сторони (або будь-якої їх комбінації), і воно може фізично існувати як всередині, так і поза юрисдикцією власника.

На сьогоднішній день існує чотири основні напрями розвитку хмарних обчислень [4]:

– iaaS;

- paaS;
- saaS;
- *aaS.

Деякі продукти безпосередньо надають користувачам такі Internet-сервіси, як системи зберігання, програмне забезпечення проміжного шару, підтримка спільної роботи і бази даних.

Інфраструктура як послуга (IaaS, англ. Infrastructure-as-a-Service) надається як можливість використання хмарної інфраструктури для самостійного управління ресурсами обробки, зберігання, мережами та іншими фундаментальними обчислювальними ресурсами, наприклад, споживач може встановлювати і запускати довільний програмне забезпечення, яке може включати в себе операційні системи, платформне і прикладне програмне забезпечення.

Споживач може контролювати операційні системи, віртуальні системи зберігання даних і встановлені програми, а також володіти обмеженим контролем за набором доступних мережевих сервісів (наприклад, фаєрволом, DNS).

Платформа як послуга (PaaS, англ. Platform-as-a-Service) - модель, коли споживачеві надається можливість використання хмарної інфраструктури для розміщення базового програмного забезпечення для подальшого розміщення на ньому нових або існуючих додатків (власних, розроблених на замовлення або придбаних тиражованих додатків). До складу таких платформ входять інструментальні засоби створення, тестування і виконання прикладного програмного забезпечення - системи управління базами даних, сполучна програмне забезпечення, середовища виконання мов програмування - надаються хмарним провайдером.

Програмне забезпечення як послуга (SaaS, англ. Software-as-a-Service) - модель, в якій споживачеві надається можливість використання прикладного програмного забезпечення провайдера, який працює в хмарної інфраструктурі і доступного з різних клієнтських пристроїв або за допомогою тонкого клієнта, наприклад, з браузера (наприклад, веб-пошта) або за допомогою інтерфейсу програми. Контроль і управління основним фізичної і віртуальної інфраструктурою хмари, в тому числі мережі, серверів, операційних систем, зберігання, або навіть індивідуальних можливостей додатка (за винятком обмеженого набору призначених для користувача налаштувань конфігурації програми) здійснюється хмарним провайдером. Приклади користувачів (бізнес-користувачі, адміністратори додатків).

Інші інфраструктури, наприклад DaaS (Desktop-as-a-Service) пропонує кожному користувачеві стандартизоване віртуальне робоче місце, з можливістю настройки і установки інших програм. Доступ здійснюється по мережі за допомогою тонкого клієнта, яким може бути що завгодно від звичайного ПК до смартфона (Google Chrome OS). CaaS (Communications-as-a-Service) - поєднання програмно-апаратних засобів для організації всіх видів спілкування (голос, пошта) між співробітниками одного підприємства за рахунок сторонніх рішень.

Альтернативний варіант SaaS просуває корпорація Microsoft, називається від S + S (Software + Services) і поєднує в собі сильні сторони типового SaaS і звичайного доступного додатки. Це звичайне ПО, але з орієнтацією на віддалені сервіси. Обчислення в хмарі перетворюються в серйозну технологічну тенденцію - багато експертів вважають, що в найближчі п'ять років cloud computing змінить не тільки IT-процеси, а й сам ринок інформаційних технологій.

Крім того, тривалий час побудови і введення в експлуатацію великих об'єктів інфраструктури інформаційних технологій і висока їх початкова вартість обмежують здатність споживачів гнучко реагувати на вимоги ринку, тоді як хмарні технології забезпечують можливість практично миттєво реагувати на збільшення попиту на обчислювальні потужності.

При використанні хмарних обчислень витрати споживача зміщуються в бік операційних - таким чином класифікуються витрати на оплату послуг хмарних провайдерів.

1.2 Хмарні обчислення - переваги і недоліки

Переваги хмарних обчислень:

– недорогі комп'ютери для користувачів. Користувачам немає необхідності купувати дорогі комп'ютери, з великим об'ємом пам'яті і дисків, щоб використовувати програми через веб-інтерфейс;

– збільшена продуктивність користувальницьких комп'ютерів. Так як більшість програм і служб запускаються віддалено в мережі Інтернет, комп'ютери користувачів з меншим числом програм швидше запускаються і працюють;

– зменшення витрат і збільшення ефективності IT інфраструктури. Звичайні сервера середньої компанії завантажені на 10-15%. В одні періоди часу є потреба в додаткових обчислювальних ресурсах, в інших ці дорогі ресурси простоюють. Використовуючи необхідну кількість обчислювальних ресурсів в "хмарі" в будь-який момент часу, компанії скорочують витрати на обладнання і його обслуговування до 50%;

– менше проблем з обслуговуванням. Так як фізичних серверів з впровадженням Cloud Computing стає менше, їх стає легше і швидше обслуговувати;

– менше витрат на програмне забезпечення, яке. Замість придбання пакетів програм для кожного локального користувача, компанії купують потрібні програми в "хмарі";

– постійне оновлення програм;

– збільшення доступних обчислювальних потужностей. Користувачі можуть запускати більш складні завдання, з великою кількістю необхідної пам'яті, місця для зберігання даних, тоді, коли це необхідно;

– необмежений обсяг збережених даних;

– сумісність з більшістю операційних систем. Доступ до програм та віртуальним комп'ютерам відбувається за допомогою веб-браузера або іншими засобами доступу, що встановлюються на будь-який персональний комп'ютер з будь-якою операційною системою;

– покращена сумісність форматів документів;

– простота спільної роботи групи користувачів;

– повсюдний доступ до документів;

– доступність з різних пристроїв;

– дружелюбність до природи, економне витрачання її ресурсів. Cloud Computing дозволяє не тільки економити на електриці, обчислювальних ресурсах, фізичному просторі, займаному серверами, а й розумно підходити до витрачання природних ресурсів;

– стійкість даних до втрати або крадіжки обладнання.

Недоліки хмарних обчислень:

– постійне з'єднання з мережею Інтернет;

– погано працює з повільним Інтернет-доступом. Багато "хмарні" програми вимагають хорошого Інтернет-з'єднання з великою пропускною здатністю;

– програми можуть працювати повільніше ніж на локальному комп'ютері;

– не всі програми або їх властивості доступні віддалено. Якщо порівнювати програми для локального використання і їх "хмарні" аналоги, останні поки програють у функціональності;

– безпека даних може бути під загрозою. Тут ключовим є слово "може". Все залежить від того, хто надає "хмарні" послуги. Якщо цей хтось надійно шифрує дані, постійно робить їх резервні копії, вже не один рік працює на ринку подібних послуг і має хорошу репутацію, то загрози безпеки даних може ніколи не статися.

Незважаючи на те, що кількість плюсів перевершує мінуси, в кожній конкретній ситуації вони мають велику важливість або навпаки, не мають ніякого значення.

Ресурси в хмарних обчисленнях об'єднані в пули і надаються на вимогу, оплата здійснюється за фактичним використанням, а сама модель відрізняється високою адаптованістю і прекрасними можливостями мережевого доступу.

Для користувачів, з точки зору обліку ресурсів, хмарні обчислення - реалізація режиму «плата на використання по вимогам», який може зручно реалізовувати доступ до спільно використовуваних ре- ресурсів IT через Інтернет.

Якщо ресурси IT включають мережу, сервер, системи зберігання, додатки, служби і т.д., вони можуть бути розгорнуті швидко і легко, з мінімальними витратами на управління і взаємодії з постачальниками послуг.

Хмарні обчислення можуть істотно поліпшити доступність ресурсів IT і дають багато премайна в порівнянні з іншими обчислювальними методами. Наприклад, вони можуть забезпечити доступ до послуг без взаємодії з постачальниками послуг. І все ресурси на хмарі доступні будь-якому користувачеві, тобто користувачі можуть динамічно орендовані фізичні або віртуальні ресурси і не повинні знати їх походження або місце проживання.

Крім того, всі ресурси на платформі хмарних обчислень можуть бути розгорнуті швидко і без зупинки обчислень. Хмарні обчислення можуть забезпечити три види режимів служби, включаючи IaaS, PaaS і SaaS. Тут SaaS означає, що послуга, надана клієнту, є прийнятна, які працюють на інфраструктурі хмарних обчислень, забезпеченої постачальниками послуг.

До них можна отримати доступ інтерфейсами тонкого клієнта, такими як браузер і т.д., PaaS дається споживачеві для розгортання на хмарної інфраструктурі, створюваної споживачами, або для запуску додатків, побудованих при використання мов програмування і інструментів, які підтримуються провайдером.

Споживач не керує і не контролює базову хмарну інфраструктуру, включаючи мережу, сервери, операційні системи або системи зберігання, але керує розгорнутими додатками і, можливо, додатком, що визначає конфігурацію операційного середовища.

IaaS пов'язаний з послугами для користувачів, які хочуть орендувати обчислювальну потужність, обсяг в системах зберігання, мережа та інші основні комп'ютерні ресурси - їх користувачі можуть розгорнути і виконати на них будь-яке програмне забезпечення, включаючи операційні системи і додатки.

При цьому користувачеві не потрібно управляти або контролювати хмарну інфраструктуру, включаючи мережу, сервери, операційну систему, системи зберігання та навіть настройки додатків.

Від точки зору розгортання платформа хмарних вимірювань включає три види якими є приватні хмари, публічні хмари і гібридні хмари.

Публічна хмара - це хмарна інфраструктура, яка доступна для широкої громадськості або співробітників корпорації і належить організації, яка надає хмарні сервіси. У публічних хмарах обчислювальні ресурси динамічно забезпечуються через Інтернет за допомогою веб-додатку або веб-служб.

Приватна хмара пов'язана з реалізацією хмарних обчислень на корпоративних мережах. Приватні хмари створені для виняткового використання однією групою користувачів, забезпечуючи повний контроль, безпеку та якість обслуговування.

Вони можуть бути побудовані і управлятися власної IT-службою компанії або постачальником хмарних сервісів. Гібридне хмара об'єднує безліч публіка- особистих і приватних моделей хмари.

Його складно визначити вичерпним чином через складність перенесення додатків в публічне хмара з приватного. Вважається, що хмарні обчислення викличуть нову революцію в IT-сервісах.

Передбачаючи величезний бізнес-потенціал такого підходу, багато країн, уряди і корпорації прийняли рішення підтримувати і вкладати кошти в розвиток методів хмарних обчислень. EC2 від Amazon, Azure від Microsoft, AppEngine від Google,

Синя хмара від IBM і т.д. - це все широко використовувані платформи хмарних вимірювань, які свідчать про те, що хмарні обчислення стануть ареною конкурентної боротьби. Однак при всьому різноманітті хмарних платформ у кожної є власні особливості та переваги, і зробити розумний вибір між ними велика проблема.

1.3 Поняття хмарного сховища даних

Хмарне сховище даних-модель онлайн-сховища, в якому дані зберігаються на численних розподілених в мережі серверах, що надаються в користування клієнтам, в основному, третьою стороною.

На противагу моделі зберігання даних на власних виділених серверах, придбаних або орендованих спеціально для подібних цілей, кількість або будь-яка внутрішня структура серверів клієнту, в загальному випадку.

Дані зберігаються, а так само і обробляються, в так званому хмарі, яке представляє собою, з точки зору клієнта, один великий віртуальний сервер.

Фізично ж такі сервери можуть розташовуватися віддалено один від одного географічно, аж до розташування на різних континентах.

Іншими словами, це своєрідний онлайн-сервіс, що надає можливість зберігати файли на віддаленому сервері. Тобто користувач може завантажити документ в будь-який онлайн-сховище і в майбутньому використовувати його прямо з сервера. З точки зору клієнта, всі операції відбуваються в одному місці, так званому «хмарі».

Однак насправді, віддалений сервер найчастіше розташовується в різних місцях, а іноді і на різних континентах. Але це анітрохи не ускладнює роботу хмарних сервісів, так як швидкість роботи залежить від клієнта. А точніше, від швидкості Інтернет - з'єднання у клієнта, яка бажано не повинна бути нижче 600 Кбіт /с.

Саме тому хмарні сервіси з'явилися зовсім не давно з причини того, що високошвидкісний інтернет з наданої швидкістю не менше 1 Мб/с. з'явився в нашій країні.

Хмарна архітектура зберігання даних - це перш за все доставка ресурсів зберігання даних на вимогу в високо масштабованому середовищі. Узагальнено хмарна архітектура зберігання даних являє собою зовнішній інтерфейс, який надає API для доступу до накопичувачів.

У традиційних системах зберігання даних це протокол SCSI, але в хмарі з'являються нові протоколи. Серед них можна знайти зовнішні протоколи Web-сервісів, файлові протоколи і навіть більш традиційні зовнішні інтерфейси (Internet SCSI, iSCSI і ін.).

За зовнішнім інтерфейсом розташовується рівень проміжного програмного забезпечення, який я називаю логікою зберігання даних. Цей рівень реалізує ряд функцій, таких як реплікація даних і скорочення обсягу даних, за традиційними алгоритмами розміщення даних (з урахуванням географічного розташування).

Нарешті, внутрішній інтерфейс організовує фізичне зберігання даних. Це може бути внутрішній протокол, який реалізує специфічні функції, або традиційний сервер з фізичними дисками.

Одна з найбільш примітних особливостей хмарного зберігання даних - здатність забезпечити економію. Це економія на придбанні накопичувачів, їх енергопостачанні, ремонті, а також на управлінні зберіганням. Якщо розглядати хмарне зберігання з цієї точки зору (включаючи SLA і підвищену ефективності зберігання), воно може виявитися вигідним при певних моделях використання.

Однією з основних завдань створення системи і способу взаємодії користувачів з хмарним об'єктним сховищем через корпоративний шлюз, розміщений в довіреної середовищі, є підвищення рівня конфіденційності даних, що зберігаються. Резюмуючи наведений вище огляд, можна виділити два основні підходи до організації хмарного об'єктного зберігання даних, що відповідає даним критеріям:

- Користувач набуває місце зберігання в центрі зберігання і обробки даних і розгортає на цьому місці власну інфраструктуру, що реалізує хмарне зберігання;
- Користувач набуває послугу зберігання даних в хмарної системі зберігання у вже існуючого сервісу.

Безумовно, з точки зору конфіденційності даних, перший підхід представляється кращим. На відміну від другого підходу, користувач не передає дані для збереження стороннім сервісам, які мають потенційну можливість для несанкціонованого використання інформації, а спрямовує їх безпосередньо службам власної програмної інфраструктури.

Проте, перший підхід також не позбавлений недоліків. З одного боку, всі дані будуть зберігатися в одному дата-центрі, що знижує рівень їх доступності та надійності зберігання в разі, наприклад, втрати зв'язку з даними дата-центром або поломки обладнання в ньому. Крім того, розгортання і підтримку власної програмної інфраструктури є економічно не вигідним, оскільки вартість її обслуговування виявиться порівнянною з вартістю обслуговування всього дата-центру.

Нижче представлена таблиця характеристик хмарної архітектури зберігання даних:

Таблиця 1.1 - Характеристики хмарної архітектури зберігання даних

Характеристика	**Опис**
Керованість	Здатність управляти системою при наявності мінімальних ресурсів
Метод доступу	Протокол, через який надаються послуги хмарного зберігання даних
Продуктивність	Вимірюється пропускною спроможністю і часом затримки
Масштабованість	Можливість поступового нарощування для задоволення нових вимог або обробки підвищеного навантаження
Готовність даних	Вимірюється часом безвідмовної роботи системи
Управління	Можливість управляти системою - зокрема, вибираючи вартість, продуктивність або інші характеристики
Ефективності зберігання	Міра ефективності використання накопичувачів

Однією з ключових характеристик хмарної системи зберігання даних є її вартість. Якщо клієнт зможе купувати і управляти ресурсами зберігання локально, а не орендувати їх в хмарі, ринок хмарних систем зберігання зникає. Його витрати можна розділити на дві загальні категорії: вартість самої фізичної екосистеми зберігання і витрати на управління нею.

Вартість управління прихована, але є довгостроковий компонент загальної вартості.

З цієї причини хмарна система зберігання повинна бути в значній мірі самокерованою. Вирішальне значення має можливість додавати нові накопичувачі, коли система автоматично переконфігурується для їх розміщення, і здатність системи знаходити і автоматично виправляти

помилки. У майбутньому такі концепції, як автономні обчислення, будуть відігравати ключову роль в хмарних архітектур зберігання.

Одним з найяскравіших відмінностей між хмарної і традиційної системами зберігання є засоби доступу до них. Більшість постачальників пропонує різні методи доступу, однак загальноприйнятими є API Web-сервісів.

Багато з них реалізовані на принципах REST, що має на увазі об'єктно-орієнтовану схему, розроблену поверх HTTP (з використанням HTTP в якості транспорту). REST-API без запам'ятовування стану прості й ефективні. REST-API реалізують багато постачальників хмарних послуг зберігання, включаючи Amazon Simple Storage Service (Amazon S3), Windows Azure ™ і Mezeo Cloud Storage Platform.

Одна проблема API Web-сервісів полягає в тому, що для того щоб скористатися перевагами хмарної системи зберігання, вони вимагають інтеграції з додатком. Тому з хмарними системами зберігання для забезпечення безпосередньої інтеграції використовуються також загальні методи доступу.

Наприклад, протоколи на основі файлів, такі як NFS / Common Internet File System (CIFS) або FTP, або протоколи на основі блоків, такі як iSCSI. Такі методи доступу надають Nirvanix, Zetta, Cleversafe і інші постачальники послуг хмарного зберігання.

Вищезазначені протоколи найбільш поширені, але для хмарного зберігання підходять і інші. Один з найцікавіших - web-based Distributed Authoring and Versioning (WebDAV). WebDAV також заснований на HTTP і дозволяє використовувати Web в якості ресурсу для читання і запису. У число постачальників, що використовують WebDAV, входять Zetta, Cleversafe і інші.

Можна знайти і такі рішення, які підтримують кілька протоколів доступу. Наприклад, IBM® Smart Business Storage Cloud дозволяє використовувати протоколи на основі файлів (NFS і CIFS) і протоколи на основі SAN в одній і тій же інфраструктурі віртуалізації систем зберігання даних.

Існує багато аспектів продуктивності, але головне завдання хмарної системи зберігання даних - це переміщення даних між користувачем і віддаленим постачальником хмарних послуг. Проблема криється в TCP, головною робочої конячці Інтернету. TCP управляє потоком даних на основі підтвердження прийому пакетів з віддаленого вузла.

Втрата або затримка пакетів призводить до застосування заходів щодо обмеження скупчень пакетів з додатковим обмеженням продуктивності щоб уникнути глобальних мережевих проблем.

TCP ідеально підходить для переміщення невеликих обсягів даних через глобальну мережу Інтернет, але не для доставки великих обсягів даних - в цьому випадку час обміну даними (RTT) збільшується.

Amazon за допомогою Aspera Software вирішила цю проблему, виключивши з рівняння TCP. Для прискорення масового переміщення даних, щоб уникнути великих RTT і великих втрат пакетів розроблений новий протокол Fast and Secure Protocol (FASP ™).

Ключем є UDP, допоміжний транспортний протокол по відношенню до TCP. UDP дозволяє вузлу управляти заторами, передаючи цей аспект протоколу прикладного рівня FASP (малюнок 1.1).

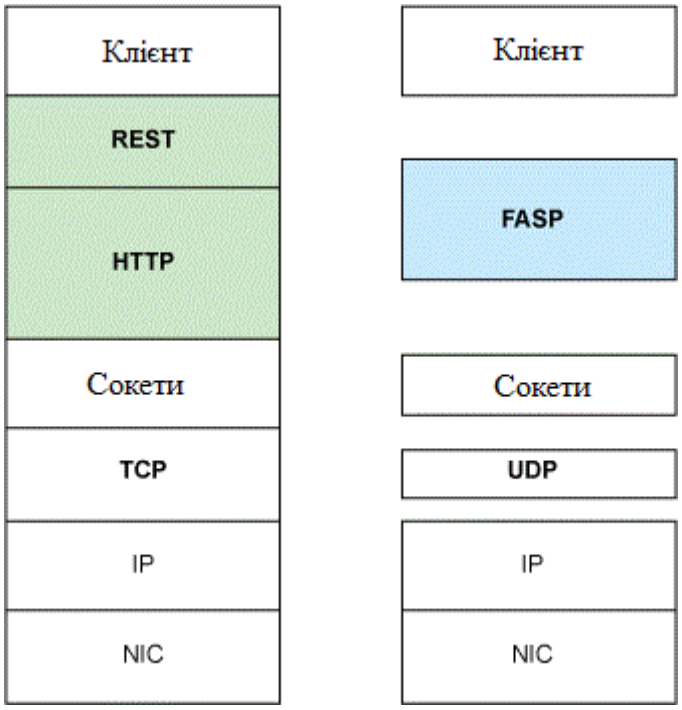


Рисунок 1.1 - Протокол Fast and Secure Protocol від Aspera Software

Працюючи зі стандартними мережевими адаптерами (без прискорення), FASP ефективно використовує доступну додатком смугу пропускання і виключає головні вузькі місця традиційних схем масової передачі даних. У розділі Ресурси наведені посилання на цікаві статистичні дані по продуктивності FASP в порівнянні з традиційними глобальними мережами, міжконтинентальними передачами і каналами супутникового зв'язку.

Масштабованість можна розглядати з кількох точок зору, але нас в основному цікавить виділення хмарних ресурсів зберігання на вимогу. Можливість нарощувати ресурси зберігання (як вгору, так і вниз) означає поліпшену економічну ефективність для користувача і підвищену складність для постачальника хмарних послуг.

Масштабованість повинна забезпечуватися не тільки для самої системи зберігання (функціональне масштабування), але і для її пропускної здатності (масштабування навантаження).

Коли постачальник хмарних послуг зберігає дані користувача, він повинен мати можливість повернути ці дані користувачеві на вимогу. З урахуванням просто мережі, помилок користувачів та інших обставин виконання цієї умови надійним і детермінованим способом може виявитися скрутним.

Існують цікаві нові схеми забезпечення високої готовності, такі як розосередження інформації. Компанія Cleversafe, яка надає послуги зберігання даних в приватному хмарі, використовує алгоритм розосередження інформації (Information Dispersal Algorithm - IDA) для підвищення доступності даних перед обличчям фізичних відмов і простоїв мережі. Алгоритм IDA, спочатку розроблений для телекомунікаційних систем Майклом Рабіном, дозволяє "нарізати" дані за допомогою кодів Ріда-Соломона для їх відновлення в разі втрати частини даних.

Важливе значення має здатність клієнта контролювати і управляти тим, як зберігаються його дані, і пов'язаними з цим витратами. Численні постачальники хмарних послуг пропонують засоби управління, які забезпечують користувачам підвищений контроль над витратами.

Amazon, щоб надати користувачам засоби мінімізації загальних витрат на зберігання даних, застосовує Reduced Redundancy Storage (RRS). Дані репліцируються в інфраструктурі Amazon S3, але RRS дозволяє репліцировати їх меншу кількість разів з можливістю відновлення в разі втрати даних. Це ідеально підходить для даних, які можна відтворювати, або коли копії даних розташовуються в різних місцях. Nirvanix також забезпечує реплікацію на основі правил, допускаючи більш детальний контроль над тим, де і як зберігаються дані.

Ефективність зберігання даних - важлива характеристика хмарної інфраструктури зберігання, особливо з огляду на її акцент на загальну економію. Наступний розділ спеціально присвячений видатках, а ця характеристика більше відноситься до ефективності використання наявних ресурсів, ніж до їх вартості.

Щоб зробити систему зберігання більш ефективної, потрібно зберігати більше даних. Спільним рішенням є скорочення обсягу вихідних даних, щоб вони займали менше фізичного простору. Два способи досягнення цієї мети: стиснення - упаковка даних шляхом їх кодування з використанням різних уявлень - і дедуплікація - виключення всіх дублікатів даних.

Хоча обидва методи корисні, стиснення передбачає обробку (перекодування даних в інфраструктуру і з неї), а дедуплікація - обчислення сигнатур для пошуку дублікатів.

Одна з найбільш примітних особливостей хмарного зберігання даних - здатність забезпечити економію. Це економія на придбанні накопичувачів, їх енергопостачанні, ремонті, а також на управлінні зберіганням.

Якщо розглядати хмарне зберігання з цієї точки зору (включаючи SLA і підвищену ефективності зберігання), воно може виявитися вигідним при певних моделях використання.

Серед постачальників хмарних послуг зберігання даних, існують хмарні моделі, які дозволяють користувачам зберігати контроль над своїми даними. Хмарне зберігання розвивається в трьох напрямках, одне з яких допускає злиття двох інших для досягнення економічної ефективності та безпеки.

Постачальники загальнодоступних хмарних систем зберігання даних, надають інфраструктуру на умовах оренди (ресурси для довгострокового або короткострокового зберігання даних і смугу пропускання мережі).

Приватні хмари використовують ті ж концепції, що і загальнодоступні, але в такій формі, що інфраструктура може бути надійно вбудована в приватну мережу користувача.

Нарешті гібридні хмарні системи зберігання дозволяють з'єднати обидві моделі, визначаючи правила, що регулюють ту, які дані необхідно зберегти в приватному володінні, а які можна захистити в рамках публічних хмар (малюнок 1.2).

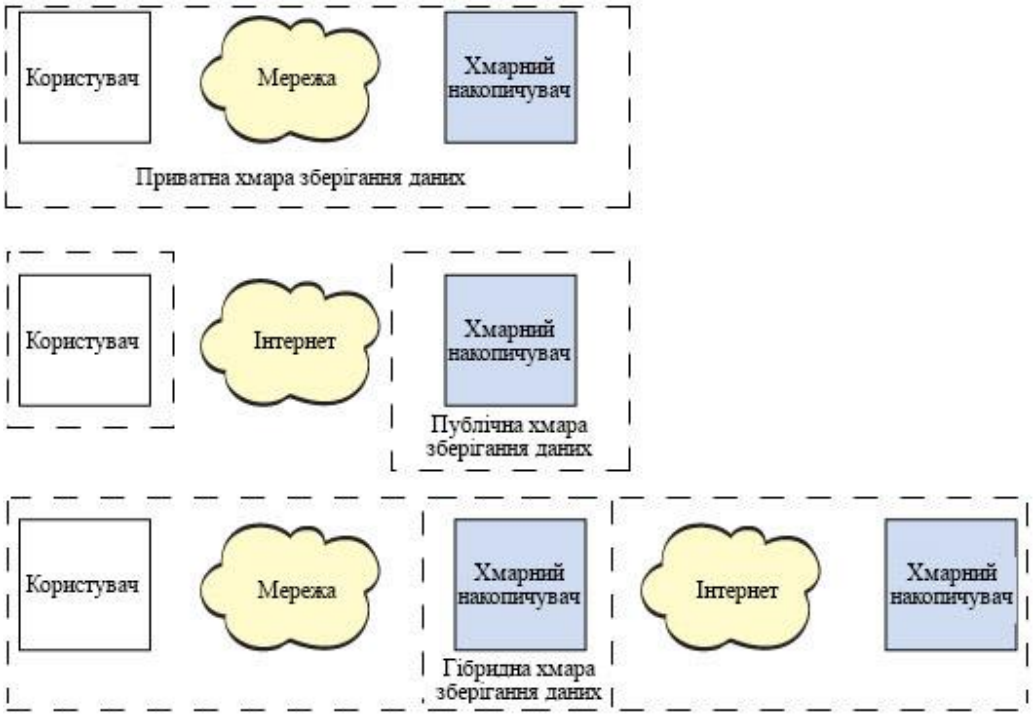


Рисунок 1.2 - Хмарні моделі зберігання даних

На малюнку 1.2 хмарні моделі показані графічно. У число постачальників загальнодоступних хмар зберігання даних входять Amazon і Nirvanix (які пропонують зберігання даних як послуги).

Прикладами постачальників приватних систем зберігання служать IBM, Parascale і Cleversafe (яка пропонує програмне забезпечення та / або обладнання для внутрішнього хмари). Нарешті, постачальники гібридних хмар - це Nirvanix, Egnyte і ін.

Такі системи є ефективним і економічно вигідним рішенням, позбавленим описаних вище недоліків традиційних підходів, яке дозволяє, з одного боку, спільно використовувати як готову інфраструктуру сторонніх сервісів хмарного об'єктного зберігання, так і приватні хмарні сховища даних, а з іншого боку - забезпечити високий рівень конфіденційності і доступності даних, що зберігається.

1.4 Висновки до розділу 1

В цьому розділі було розглянуто поняття хмарних обчислень, ресурсів та середовища загалом. На сьогоднішній день існує чотири основні напрями розвитку хмарних обчислень – це IaaS, PaaS, SaaS та *aaS.

Проаналізовані переваги і недоліки хмарних обчислень, їх сильні та слабкі сторони. Хмари поділяються на приватні, публічні, гібридні і кланові. Кожна з них є інфраструктурою, яка може перебувати у власності, управлінні та експлуатації.

Також було виділено декілька видів хмар, такі як SAAS, PAAS, службові обчислення, веб-сервіси в «хмари», MSP та комерційні платформи для сервісів.

Розглянуті сучасні принципи побудови хмарних ресурсів.

В ході аналізу було виділено наступні характеристики хмарної архітектури зберігання даних: керованість, метод доступу, продуктивність, масштабованість, готовність даних, управління та ефективності зберігання.

Поставлені такі задачі:

– проаналізувати існуючі на даний момент протоколи доступу до середовища хмарних ресурсів. Скласти єдиний критерій оцінки ефективності протоколів доступу до середовища хмарних ресурсів, в одному інтервалі часу.

– визначити тестове завдання і методику тестування. Порівняти час виконання основних етапів тестового завдання і аналіз навантаження на процесор, оперативну пам'ять і мережу. Зробити висновки за результатами тестування.

– розглянути параметри і характеристики, що безпосередньо впливають на підвищення ефективності протоколів доступу до середовища хмарних обчислень.

– розробити програмне забезпечення, завданням якого є автоматичне тестування протоколів, з можливістю вибору тестових завдань різної складності і характеристик.

РОЗДІЛ 2
АНАЛІЗ СИСТЕМ ВІРТУАЛІЗАЦІЇ І ЗВ'ЯЗКУ МІЖ КОМПОНЕНТАМИ

2.1 Основні принципи побудови систем віртуалізації

При проектуванні і побудові систем віртуалізації важливо враховувати необхідність забезпечення надійної та відмовостійкості роботи обладнання та сервісів. Досягається це в першу чергу за рахунок резервування обладнання та кластеризації.

Кілька однакових по характеристикам серверів (важливо збіг серій процесорів, материнських 19 плат, а також типу і кількості мережевих інтерфейсів) за допомогою програмних засобів об'єднують в єдиних кластер. При цьому на кожному з серверів-вузлів кластера встановлюється агент, який сповіщає центральний вузол про свою працездатності і передає докладні дані про завантаженість обчислювальних ресурсів. Деякі програмні продукти дозволяють в автоматичному режимі відслідковувати та здійснювати живу міграцію віртуальних машин, запущених на сервері, між іншими вузлами кластера, завдяки цьому досягається більш рівномірний розподіл обчислювальної навантаження і зменшується ймовірність виконання збою в роботи системи.

Кожен фізичний сервер - гіпервізор обладнаний як мінімум подвійним набором мережевих пристроїв (LAN Ethernet комутаторів, Fiber Channel комутаторів і ін.). Таке дублювання також диктується вимогою виключити фактів відмови з боку мережі і максимально знизити можливість мережевий ізоляції гіпервизора від інших вузлів кластера. Крім резервування серверів, відмовостійкість сервісів досягається також при правильному проектуванні і налаштування систем зберігання даних. Основним фактором, що допомагає значно збільшити надійність і продуктивність систем віртуалізації є фактор распределенности доступу до СГД. Для досягнення распределенности доступу файли віртуальних машин, образи і шаблони операційних систем зберігають не на локальних дисках, підключених до гіпервізора, а на мережевих файлових системах.

Інформація, що зберігається на мережевих СГД резервується не тільки за рахунок логічного розбиття дискового простору (формування RAID масивів), в них застосовуються складні алгоритми розподілу і резервування даних, такі алгоритми пропрієтарних і унікальні для кожного виробника СГД. Їх застосування багаторазово збільшує швидкість звернення до даних, знижуючи ймовірність утворення ефекту «пляшкового горлечка »при доступі гіпервизора до даних віртуальної машини і захищає від можливих збоїв записи і пошкодження даних. Сучасні СГД, використовувані для побудови систем віртуалізації, мають високопродуктивні мережеві інтерфейси Fiber Channel (більше 8 Гбіт / с) або iSCSI (більше 1 Гбіт / с).

Виконання всіх рекомендацій призводить до сценарію, при якому кожен гіпервизор повинен бути пов'язаний з іншим гіпервізором-партнером або системою зберігання як мінімум двома шляхами, щоб незалежно від причин, в випадку розриву з'єднання, завжди залишався резервний шлях, і система не втратила функціональності, продовжуючи надавати послуги користувачам. Тонка суцільна лінія позначає з'єднання між SAN комутаторами, встановленими в Гіпервізор, і SAN комутаторами, жирна суцільна лінія показує з'єднання між СГД та SAN комутаторами, жирна переривчаста з'єднання між LAN комутаторами гіпервизора і LAN комутатором.

Застосування WaaS моделі надання послуг (надання робочих станцій в якості послуги) в хмарних системах стає все більш популярним рік від року, пов'язано це з підвищенням інтересу до впровадження інфраструктури віртуальних робочих столів з боку навчальних, медичних закладів та промислових підприємств.

При цьому користувач не дбає про установці і ліцензування операційної системи та іншого прикладного програмного забезпечення, необхідного йому для роботи, а лише займається їх експлуатацією. VDI є складною обчислювальною системою, що надає користувачам послуги відповідно до моделі WaaS, що складається з декількох компонент: сервер віртуалізації (гіпервізор) - брокер підключень - служба каталогів Active Directory - віртуальний робочий стіл - тонкий клієнт - VDI протокол.

Сервер віртуалізації є фізичний сервер, що знаходиться в віддаленому ЦОД з підключеною підсистемою зберігання даних і встановленим ПО віртуалізації - гіпервізором, який дозволяє створювати віртуальні машини, контролювати і розподіляти обчислювальні ресурси сервера між ними. Брокер підключень - віртуальний сервер основним завданням якого є підбір підходящої віртуальної машини (ВРС) для користувача в Залежно від наданих їм облікових даних.

Система каталогів Active Directory є службою контролю доступу до ресурсів інфраструктури ВРС. Завдяки службі каталогів створюється єдина точка авторизації і автентифікації, необхідна для централізації управління бізнес процесами. Вона являє собою LDAP систему компанії Microsoft, що зберігає в LDAP базі даних інформацію про облікові записах користувачів, комп'ютерів, груп, принтерів та ін., а також що дозволяє використовувати централізоване застосування групових політик на об'єкти користувачів і комп'ютерів.

Віртуальний робочий стіл - віртуальна машина (операційна система) з перед встановленим спеціалізованим ПО (VDI агентом), що знаходиться на гіпервізорі. Тонкий клієнт - без дисковий обчислювальний модуль з апаратної або програмної підтримкою підключення до віртуального робочого столу (VDI клієнта) через брокер підключень. VDI протокол - алгоритм, завдяки якому користувач з тонкого клієнта підключається до ВРС і передає в режимі реального часу

зображення з робочого столу віртуальної машини, яка працює віддалено в хмарі. VDI протокол являє собою клієнт-серверний додаток, агент (серверна частина) встановлюється на віртуальній машині, а клієнт на стороні користувача.

Клієнт ініціює з'єднання, авторизуючи на службових серверах VDI і підключається до агента на цільової віртуальній машині. Існує кілька компаній-виробників програмних рішень для VDI і кожне рішень має свою реалізацію для VDI протоколів. Найбільшими і найбільш популярними виробниками в Зараз є компанії Microsoft з термінальним протоколом RDP (Remote Desktop Protocol) і VMware з протоколом PCoIP (Personal Computer over Internet Protocol). Інфраструктура ВРС дозволяє вносити зміни в «апаратну» і програмну частини без «відключення живлення» віртуальної машини, не перериваючи роботи користувача. Перелік всіх доступних віртуальних робочих столів, а також доступних шаблонів і підключених користувачів доступний в єдиної консолі, що володіє високим ступенем інформативності. Всі ці плюси доповнюються зниженням витрат на обслуговування інфраструктури в порівняно з класичною інфраструктурою, коли кожному користувачеві виділяється повноцінний «товстий» системний блок і основні обчислювальні ресурси розміщуються не в «хмарі», а локально.

2.2 Протоколи доступу до середовища хмарних обчислень

Всі запропоновані в дипломній роботі методи і тести запускалися на найпоширеніших VDI протоколах - RDP і PCoIP. VDI рішення інших виробників, серед яких HewlettPackard і Citrix, не розглядалися, оскільки вони мають менший популярності. RDP (Remote Desktop Protocol - протокол доступу до віддаленого робочого столу) - закритий протокол прикладного рівня (рівень моделі OSI), придбаний корпорацією Microsoft у компанії Citrix, який використовується для забезпечення віддаленого доступу користувача з сервера або робочої станції, на якому розгорнута служба термінальних підключень. RDP клієнти написані практично для всіх існуючих сучасних операційних систем (Windows, Macintosh, Linux, * UNIX) і за замовчуванням присутні в системах сімейства Windows. За замовчуванням агент (серверна частина) прослуховує для підключення порт TCP 3389. Офіційна назва ПО для клієнтського доступу - remote Desktop Connection або Terminal Services Client (TSC).

Основні характеристики:

– підтримка 32 - бітного кольору (додатково до підтримуваним в ранніх версіях 8-, 15-, 16-, і 24-бітного);

– можливість 128 - бітове шифрування, що використовує алгоритм шифрування RC4 (значення безпеки за замовчуванням, старі клієнти можуть використовувати більш слабке шифрування);

– підтримка протоколу Transport Layer Security (TLS);

– переведення звуку з віддаленого ПК і відтворення на локальному комп'ютері;

– підключення локальних ресурсів до віддаленої машині (Мапування);

– використання локальних або мережевих принтерів на віддаленому ПК;

– надання можливості додатків, що виконуються в рамках поточного сеансу, звертатися до локальних портів;

– можливість використовувати буфер обміну для обміну інформацією.

PCoIP (Personal Computer over Internet Protocol - персональний комп'ютер по протоколу IP, «ПК-через-IP») - пропрієтарний протокол, який використовується в рішеннях, пов'язаних з віддаленими робочими станціями і робочими столами. Спроектований і розроблений компанією Teradici, надалі просувається компанією VMware, що інтегрує його в свої продукти для доступу до консолям віртуальних машин або віддалених робочих столів VDI інфраструктури.

Технологія PCoIP дозволяє здійснювати віддалений доступ до високопродуктивним робочих станцій в ЦОД з "нульового клієнта" (або нуль-клієнта) PCoIP Zero Client або LCD-монітора з вбудованим клієнтським процесором PCoIP. Протокол PCoIP також вбудований в VMware Horizon View для забезпечення віддаленого доступу до віртуальних машин При використанні PCoIP ПК умовно розділяється на дві частини (Системний блок і периферійні пристрої) і вони з'єднують за коштами IP протоколу (тому його і називають PC-over-IP). Системний блок (Обчислювальні потужності) розташовується в ЦОД, а периферійні пристрої підключаються безпосередньо на майданчику користувача через різні порти на тонкому клієнті.

Спочатку були задумані два типи реалізації рішення віддаленого доступу до ВРС за допомогою PCoIP - програмна і апаратна. Програмна реалізація представляє собою класичну схему роботи клієнт-сервер, в даному сценарії і клієнтська, і серверна частини додатки виконані у вигляді ПО і для обробки процесів PCoIP використовуються ресурси самої віртуальної машини, а точніше ресурси ЦОД в якому вона розташовується. Апаратна реалізація передбачає, що на клієнтської і / або серверній стороні обробка процесів PCoIP відбувається в спеціалізованому апаратному чипі. Після продажу свого протоколу компанії VMware, Teradici розробляє тільки над удосконаленням апаратних рішень PCoIP, яка передбачає установку спеціальних процесорів на клієнті і на сервері. Існують пристрої з підтримкою PCoIP від сторонніх виробників, серед таких пристроїв дисплеї зі вбудованими процесорами PCoIP, настільні нуль-клієнти (PCoIP Zero Clients) і серверні плати розширення.

Технологія PCoIP дозволяє здійснювати віддалений доступ до робочих столів, розгорнутих на комп'ютерах в центрах обробки даних з широкого спектра споживчих пристроїв: від персональних комп'ютерів, ноутбуків, тонких клієнтів, планшетів, мобільних телефонів, на які встановлюється спеціалізоване клієнтське програмне забезпечення або моніторів, «нульових

клієнтів »- пристроїв з вбудованим апаратним процесором PCoIP. На стороні робочих станцій підтримується як апаратна, так і програмна реалізація захоплення робочого столу. Використовується в програмному забезпеченні з організації віртуальної інфраструктури робочих столів, зокрема, підтримується в VMware View, забезпечуючи віддалений робочий стіл до віртуальних машин. Процесори, що забезпечують апаратне рішення по захопленню робочих столів і їх відображення на термінальних пристроях, Teradici розробляє самостійно, програмні реалізації за ліцензією випускають інші компанії [1].

Протокол PCoIP передбачає стиснення, шифрування і кодування інформації про екранному буфері, забезпечує передачу PCoIP-пристроїв тільки даних із зміненими пікселях. Підтримується передача зображення високої чіткості і навіть більш високої роздільної здатності, частота кадрів, придатна для роботи з тривимірною графікою, сумісність з інтерфейсами USB. Вимоги до пропускної здатності мережі при використанні PCoIP варіюються від 200 Кбіт / с для простих робіт, 1 Мбіт / с при інтенсивній роботі з офісними документами і веб-серфінгу і до 54 Мбіт / с при роботі з тривимірною графікою у високому дозволі.

Протокол PCoIP стискає і шифрує весь опрацьований потік даних в ЦОД і за стандартною IP-мережі передає інформацію очікують PCoIP- пристроїв. При цьому відбувається передача лише інформації про змінилися пікселях. Технологія PCoIP дозволяє здійснювати центральне управління корпоративними ПК і робочими станціями, при цьому забезпечуючи передачу зображення високої чіткості (Full High Definition) або навіть більш високого дозволу, відповідну частоту кадрів для 3D-графіки і HD-медіа, повну сумісність з інтерфейсом USB, локальну роботу по ЛВС або віддалену по ГВС. Вибрані протоколи мають широкий діапазон можливостей, серед яких відтворення відео і різних графічних ефектів, шифрування даних, контроль якості картинки і багато іншого, але вони значно відрізняються за своєю суттю. Також протокол працює шляхом рендеринга клієнтських робочих столів на мережевому або хмарному сервері. Потім пікселі робочого столу стискаються, шифруються і передаються на клієнтський пристрій. Потім клієнт розшифровує і відображає подання робочого столу для користувача.

PCoIP використовує протокол UDP (User Datagram Protocol), альтернативу протоколу TCP (Transmission Control Protocol). Протокол UDP підтримує зв'язок між процесами і використовує стійкі до втрат з'єднання з низькою затримкою для зв'язку програми з інтернетом. PCoIP надає растрові зображення, які визначають, де піксель з'являється на екрані і якого кольору він повинен бути, шляхом кодування їх на віддаленому хості, а потім потокової передачі даних клієнта. Він передає тільки області екрану, які змінюються від кадру до кадру. З точки зору кінцевої точки, це майже так, як якщо б клієнт дивився фільм в режимі реального часу про дії робочого столу. Протокол може обробляти мультимедійні та графічні додатки, але він збільшує навантаження на

процесор віддаленого хоста. Протокол PCoIP компанії Teradici створений і дебютував в 2007 році.

Спочатку, PCoIP був апаратним продуктом віртуалізації робочих столів, який будується навколо сервера, зображення робочого столу, а також клієнтського пристрою. Клієнтський пристрій було оснащено фірмовим чіпом, який дозволяв використовувати зв'язок PCoIP між клієнтом і сервером. Також PCoIP залежав від патентованого обладнання, але в кінцевому рахунку Teradici створив програмну версію протоколу PCoIP. Компанія Teradici ліцензувала програмне забезпечення VMware в 2008 році. VMware використовував PCoIP для доставки віртуальних робочих столів з тим, що тоді називалося VMware View, тепер Horizon.

PCoIP краще всього відомий своїм використанням в VMware Horizon View, але Teradici продовжує пропонувати карти віддалених робочих станцій, нульові клієнти, графічні агенти і клієнти, які дозволяють організаціям використовувати PCoIP в центрах обробки даних і хмарних розгортання.

Основна перевага PCoIP полягає в тому, що вона дозволяє організації уникати використання комп'ютерів як кінцевих точок клієнтів, покладаючись замість цього на тонкі клієнти або нульові клієнти. Тонкі клієнти і нульові клієнти коштують ніж ПК і не вимагають майже ніякого обслуговування. Крім того, тонкі клієнти і нульові клієнти, як правило, більш безпечні, ніж ПК, оскільки вони не залежать від локальної операційної системи.

І навпаки, PCoIP залежить від мережі або хмарного сервера для візуалізації робочих столів. В результаті, якщо виникне проблема з відображенням на стороні сервера, це може вплинути на кожного користувача віртуального робочого столу. Однією з найбільш поширених проблем з PCoIP є затримка. Додатки з інтенсивною графікою можуть відображатися неправильно, якщо пропускна здатність недостатня або якщо сервер грузне в запитах користувачів.

Основна відмінність полягає в характері роботи на четверному (транспортному) рівні еталонної моделі OSI. Під час своєї роботи протокол RDP використовує TCP трафік, встановлюючи з агентом з'єднання за допомогою механізму тристороннього рукостискання. Використання TCP дозволяє домогтися з'єднання з гарантованою доставкою мережевих пакетів і в разі втрати в мережі даних запросити повторну передачу даних. Протокол PCoIP використовує в своїй основі мережевої протокол UDP - протокол без гарантованої доставки даних, без повторної пересилання втрачених даних.

У статті [14] представлені результати практичного виміру продуктивності Web сервісів .Net, розгорнутих в середовищі Cloud Computing - Microsoft Azure. При проведенні експериментів використовувалася методика вимірювань, яка дозволяє відокремити характеристики самого web

- сервісу від затримок, внесених середовищем комунікації Інтернет. Отримані результати свідчать про значну невизначеності характеристик web - сервісів, розгорнутих в Cloud - середовищі і що викликаються за допомогою глобальної мережі Інтернет. Показано, що технології розробки і розгортання web - сервісів суттєво впливають на їх продуктивність.

Необхідність експериментального дослідження продуктивності web - сервісів і систем Cloud Computing обумовлена тим, що апріорне прогнозування швидкодії web - додатків, розгорнутих в Cloud - середовищі утруднено незважаючи на те, що провайдери Cloud - платформ призводять апаратні характеристики надаються віртуальних машин.

Різниця в технологіях реалізації сучасних web додатків, а також нестабільність характеристик середовища взаємодії Інтернет істотно впливають на продуктивність web додатків і обумовлюють значну ступінь невизначеності при їх практичному вимірі.

У той же час, експериментальні дослідження, подібні викладеним в статті, представляються важливими, оскільки надають розробникам і користувачам web додатків інформацію для вибору між технологіями їх реалізації і способами розгортання, а також прогнозування не функціональних характеристик.

В роботі [20] проводилось дуже цікаве порівняння протоколів доступу до віртуальних ПК і додатків в інфраструктурі VMware Horizon 7. Автор порівнював продуктивність перевіреного часом PCoIP і прийшов йому на зміну протоколу Blast Extreme в наступній тестовій конфігурації:

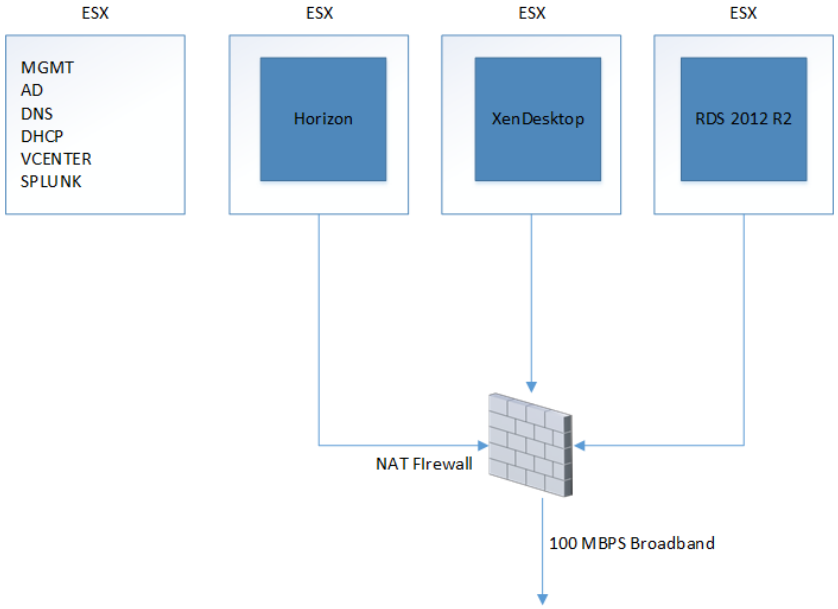


Рисунок 2.1 Тестова конфігурація інфраструктури VMware Horizon 7

Автор звертає увагу на те, що PCoIP працює по UDP, тому схожий на гоночну машину (найкраще поводиться на широкій смузі каналу без перешкод і високого навантаження), а Blast

Extreme, що працює по TCP - це джип, який добре їде по пересіченій місцевості (тобто, адаптується до параметрів каналу).

Тестування проводилося за наступною схемою:

– користувач авторизується і чекає 1 хвилину, щоб сесія налаштувалася і була готова до тестування;

– відкрили локальний PDF - файл, Скролл його вгору і вниз 1 хвилину;

– зайшли на новинний сайт з графікою http://www.vg.no і поскролілі його;

– відкрили Word і друкували там лабуду протягом 1 хвилину;

– відкрили трейлер фільму Captain America Civil War в повний екран браузера Chrome на повну тривалість (2 хвилини).

Для збору даних і налаштування оточення використовувалися наступні засоби:

– splunk - Uberagent (збір даних);

– netbalancer (bandwidth, можливість установки параметра packet loss, визначення лімітів по bandwidth limits і завдання latency).

Автор робить висновок, що PCoIP дає набагато більше навантаження на клієнтський пристрій, ніж Blast Extreme, який, в свою чергу, дає кращий User Experience, але споживає велику ширину каналу. Це може бути пов'язано з додатковими накладними витратами на квитанції TCP, а також тим, що Blast Extreme тестує канал при початку передачі і намагається вичавити з нього максимум. Blast вміє використовувати широкий канал і дає кращий user experience, а також створює менше навантаження на клієнтський пристрій.

В публікації [19] автор порівнює одразу три протоколи RDP, HDX та ICA.

Протоколи віддаленого дисплея мають свої обмеження, особливо якщо мова йде про доставку графічно-інтенсивних програм. Велика продуктивність вимагає великої пропускної здатності, яка може засмітити мережу. Крім того, якщо ви хочете використовувати низький рівень використання центрального процесора, ваш протокол підвищить пропускну здатність і послабить продуктивність кінцевого користувача. Експерти комп'ютерної віртуалізації кажуть, що ви можете вибрати будь-які два з наступних:

– низька смуга пропускання;

– хороший досвід;

– низький процесор.

У минулому RDP мала слабку продуктивність у графічному, спалахувальному та рухомому об'єкті порівняно з ICA. В результаті Microsoft внесла безліч удосконалень у протокол

віддаленого робочого стола за допомогою RemoteFX. Справді, сьогодні в середовищі локальної мережі важко визначити продуктивність у віртуальних додатках, порівняно з Citrix HDX-ICA.

Чисте середовище Microsoft RDS офіційно підтримує клієнти Windows і Mac для RDP. Citrix має клієнти ICA (тепер їх називають Citrix Receivers) для Windows, Mac, Linux, BlackBerry, Android, iOS та HTML5. Однак такі компанії, як Parallels, пропонують сторонньому клієнту, здатному покращити продуктивність RDP над будь-якою операційною системою.

З Windows, як і раніше, є переважною ОС для багатьох людей, RDP, здається, досить хороший варіант для декількох підприємств. Недоліком RDP є те, що продуктивність за низькою пропускною спроможністю WAN вимагає додаткової оптимізації, щоб забезпечити задоволення користувацького досвіду. З іншого боку, HDX - ICA, здається, має невелику перевагу зі своїми розширеними функціями. Тим не менш, дорога цінова етикетка HDX та її складності є для багатьох турботами. Не всі підприємства можуть дозволити собі дорогу цінову етикетку Citrix. Використовуючи звичайні додатки для продуктивності, ви не зможете побачити різницю між цими двома параметрами. Тим не менш, під час використання потокового мультимедіа та багатих графіків, ICA має невелику перевагу в бій RDP проти ICA.

Автор робить висновок, що протягом останніх кількох років RemoteFX RDP значно покращився, покращуючи враження від користувачів та продуктивність, близькі до найвідомішої HDX-ICA. Хоча практично неможливо помітити різницю в середовищі локальної мережі, у низькій пропускній здатності, WAN RemoteFX RDP як і раніше потребує оптимізації для досягнення якісної доставки HDX-ICA.

Microsoft RDP офіційно підтримує лише Windows і Mac, однак, Parallels компенсує цей розрив, пропонуючи широкий діапазон клієнта RDP, що дозволяє вам надавати віртуальне додаток на будь-якому пристрої. Microsoft RDP, підтримуваний економічно ефективним способом зв'язку між брокерськими програмами, такими як Parallels RAS, є більш актуальним в будь-який час, коли організація прагне до високої продуктивності, зниження вартості та швидкого рентабельності інвестицій.

У цій главі проводиться огляд факторів, що впливають на продуктивність хмарної інфраструктури, наводяться методи, використовувані в даний час для визначення продуктивності хмарних високопродуктивних систем.

Показуються складності з якими стикаються при оцінці продуктивності протоколів доступу до хмарної інфраструктурі і обґрунтовується необхідність створення нової методики, позбавленої цих недоліків.

Основна увага приділяється розробці нової моделі оцінки продуктивності протоколів доступу до віртуальних робочих столів і методикою тестування, що дозволяє зв'язати воєдино різні компоненти продуктивності. Показується як з використанням статистичних методів і

застосовуючи розподіл Зіпфа можна пов'язати такі компоненти продуктивності як продуктивність процесора і оперативної пам'яті.

При експлуатації ВРС, користувач змушений працювати в «хмарі», всі його запити обробляються віддалено в мережі, з використанням обчислювальних ресурсів ЦОД.

Оскільки з'єднання з віддаленою віртуальною машиною відбувається через мережний зв'язок, то якість цього з'єднання (рівень втрат і затримок) і пропускна здатність мережі стають одними з критичних чинників, від яких буде залежати задоволеність користувача.

Наприклад, при недостатньо широкому каналі передачі даних на моніторі клієнта буде спостерігатися затримка в відображенні зображення, а при високому рівні втрати пакетів з'єднання клієнта з ВРС може і зовсім розірватися. Всі ці негативні фактори знижують якість надаваних користувачеві послуг, а значить можуть зробити використання клієнтів не доцільним.

Крім мережевих ресурсів, для забезпечення високого рівня сервісу, наданого користувачеві, важливо знати яка кількість обчислювальних ресурсів потрібно виділити для його віртуальної машини.

Основними обчислювальними ресурсами, що витрачаються в ВРС і впливають на його продуктивність є кількість і частота віртуальних процесорів, а також кількість і частота оперативної пам'яті. Брак будь-якого з цих ресурсів призведе до того, що розподіляється між усіма додатками кількість ЦПП або ОЗП може знизитися на стільки, що його буде недостатньо для підтримки з'єднання ВМ і клієнта або ж істотно сповільниться робота операційної системи.

Не залежно від причин уповільнення, користувач потрапляє в ситуацію, коли його подальша робота стає неможливою. Calyam в своїх роботах використовує для формування оцінки задоволеності користувача показника QoE, що показує якість користувальницьких вражень від роботи. QoE сильно знижується при нестачі як процесорних ресурсів, так і ресурсів оперативної пам'яті.

Важливо також зазначити, що крім апаратних ресурсів, що виділяються на кожну віртуальну машину, часто, більш важливим фактором для користувача стає продуктивність VDI протоколу хмарних середовищ, використовуваного для підключення.

Завантаження обчислювальних ресурсів і ефективність роботи термінальних протоколів безпосередньо залежить від завдання, що виконується користувачем під час роботи за ВРС. Наприклад, якщо він працює з текстовим редактором, то можна виділити із загального споживання наступні складові:

1) споживання ОЗП:

– кількість пам'яті, що вимагається для завантаження операційної системи, в рамках якої відбувається обробка дій користувача і інших сервісних (супутніх) підпрограм, що підтримують загальну працездатність операційної системи;

– кількість пам'яті, необхідний для запуску і подальшої роботи цільового додатки (в даному випадку текстового редактора);

– кількість пам'яті, достатньої для роботи агентської (або серверної) частини термінального протоколу.

2) завантаження ЦПП:

– кількість тактів, необхідне для завантаження ОС і сервісних служб;

– кількість тактів, необхідне для запуску і обробки процесу введення символів, форматування і т.д;

– кількість тактів, необхідне для роботи серверної частини термінального протоколу.

3) завантаження ШПД:

– мінімальна пропускна здатність мережі, яка потрібна для підтримки підключення між клієнтською частиною термінального протоколу, встановлюється на клієнті і серверної, яка встановлюється на віртуальній машині, до якої власне, і йде підключення;

– пропускна здатність мережі, необхідна для передачі на монітор клієнта зображення робочого столу віртуальної машини (В даному випадку при появі і перенесення робочих вікон на столі, форматуванні тексту)

Якщо ж завдання, що виконується користувачем зміниться, для прикладу він почне переглядати відео ролик, то зміниться і споживання ресурсів.

Завдяки тому, що віртуальний робочий стіл отримує постійні масштабні, щодо попередньої задачі, поновлення, то і VDI протокол, для обробки цього завдання повинен буде забирати під свою роботу більше ресурсів.

Різко зросте споживання ЦПП і ОЗП, збільшиться мережевий трафік між клієнтом і віртуальною машиною, а це може вплинути на комфорт сприйняття користувачем свого робочого місця.

В умовах, коли кількість віртуальних і мережевих ресурсів, що виділяються на одну віртуальну машину обмежена, можна говорити про те, що рівень комфорту користувача, що працює у віртуальному середовищі, буде залежати від продуктивності VDI протоколу.

Кожен з ресурсів, споживаних при виконанні підключення до ВРС, вимірюється в власний величинах, наприклад, тактова частота процесора - Гц, обсяг оперативної пам'яті - МБ, швидкість передачі даних по мережі - біт/с і т.д. Сучасні дослідники працюють над визначенням продуктивності в VDI системах виробляють порівняння по кожному з ресурсів.

P. Calyam в своїх роботах оцінював продуктивність VDI протоколів залежно від операцій, які виконував користувач на віртуальній машині, таких як відкриття інтернет-браузера, виконання розрахунку в середовищах MathCad і MathLab, перегляд відео роликів і робота з

офісними додатками. Було сказано, що для кожного алгоритму користувача повинен бути складений профіль споживання ресурсів.

У рамках кожного з профілів дослідники пропонували використовувати 3D діаграми, згідно яких по кожній з трьох осей відкладалося значення споживаного ресурсу. В якості основних ресурсів, споживаних додатком при роботі через VDI протоколи, були розглянуті обсяг споживаної ОЗП, ЦПП і завантаженість каналу передачі даних між клієнтом і віртуальною машиною. Залежно від завдання (профілю) будувалися діаграми продуктивності для кожного з досліджених протоколів.

Така інтерпретація уможливила наочне уявлення профілів споживання у вигляді діаграм, дозволила значно спростити сприйняття і прискорити час прийняття рішення по кожному протоколу в залежності від поведінки графіків для кожного профілю.

2.2.1 Методи оцінки продуктивності систем віртуалізації

У міру розвитку обчислювальних систем і появи технологій розпаралелювання завдань між декількома обчислювальними вузлами, все більша кількість завдань стала вирішуватися з використанням високопродуктивних систем.

Виникла потреба у визначенні продуктивності обчислювальних комплексів і створенні механізму, який дозволив би максимально об'єктивно порівнювати ВПС відносно один одного. Для визначення продуктивності обчислювальних систем було розроблено безліч тестів, серед яких тести Whetstone, Dhrystone і LINPACK.

Тест Whetstone був запропонований командою дослідників H. J. Curnow і B. A. Wichmann з Британської національної фізичної лабораторії в 1976 р. Вони презентували набір програм для вимірювання продуктивності обчислювальних систем. Для написання програм використовувалася мова програмування Algol-60. Це був перший опублікований інструмент для проведення контрольно-вимірювальних тестів продуктивності. Тест Whetstone вдає із себе набір синтетичних тестів, розроблених з використанням експериментальних статистичних даних розподілу інструкцій проміжного рівня компілятора Whetstone Algol.

Набір тестів Whetstone складається з декількох модулів, програмно імітуючи навантаження для найбільш часто зустрічаючих режимів виконання обчислювальних задач (цілочисельні обчислення - integer, обчислення з плаваючою точкою - float, оператори типу «якщо» - «if», виклики функцій і т.д.).

Виконання кожного модуля проводиться з багаторазовими повтореннями, згідно вихідної Whetstone-інструкції. Здійснюється це за допомогою використання операторів циклу з різною кількістю повторень (від 12 до 899) всередині яких розміщуються програмні модулі, а

продуктивність розраховується як відношення числа Whetstone-інструкція до загального часу виконання всіх модулів набору і вимірюється в KWIPS або MWIPS.

Інструкції не прив'язані до системи команд будь-якої конкретної ОС, тобто оцінка, яка виробляється в MWIBS, є моделенезалежною. Існує значна кількість версій тесту Whetstone, проте єдина зареєстрована Британським товариством стандартів BSI-QAS написана на мові Pascal. Інші версії, написані з використанням FORTRAN, причому компанії-виробники можуть вносити зміни в тест, оптимізуючи оцінку Whetstone.

Тест Dhrystone в значній мірі схожий з тестом Whetstone і заснований на типовому розподілі мовних конструкцій. Він складається з 12 модулів, кожен з яких відповідає за конкретний типовий режим обробки програм. За допомогою тестів Dhrystone оцінюють не продуктивність всієї обчислювальної системи в цілому, а лише її окремо взятого системного або прикладного ПЗ.

У тесті Dhrustone використовується набір зі ста команд, з яких 53 оператора присвоювання, 32 команди управління, 15 викликів функцій. В якості одиниць виміру застосовується Dhrystone в секунду. Цикли тесту являють собою типовий набір фрагментів програм на мові FORTRAN.

У цих програмах реалізовані різні обчислювальні алгоритми: сіткові, хвильові, послідовні. Їх вибір був заснований на багатому досвіді створення суперкомп'ютерів і проведення складних наукових і інженерних розрахунків Ліверморської національної лабораторії ім. Лоуренса (Lawrence Livermore National Laboratory, LLNL) Міністерства енергетики США.

Коефіцієнт розпаралелювання застосовуваних алгоритмів лежить в діапазоні від 0 до 1. Це дозволяє використовувати "Ліверморської цикли" для оцінки продуктивності обчислювальних систем, що мають різну архітектуру. В даний час тест виду Dhrystone практично не використовується для оцінки продуктивності високопродуктивних систем.

Ще одним варіантом тестування є тестування за допомогою програмних бібліотек LINPACK. В основі використовуваних в LINPACK алгоритмів лежить метод декомпозиції, широко застосовуваний при високопродуктивних обчисленнях. Перевагою тестів LINPACK є їх структурованість. Для реалізації елементарних операцій над векторами, які включають множення векторів на скаляр, додавання векторів, скалярний добуток векторів виділяється базовий рівень системи, званий BLAS. Вихідні дані для тестування представляються у вигляді дійсних чисел подвійної точності. Отримані результати виражаються в FLOPS, які показують, скільки операцій з плаваючою комою виконується комп'ютером в секунду.

Тест LINPACK має два рівні. У тесті першого рівня LINPACK DP використовується вихідна матриця розміром 100 * 100. У тесті другого рівня LINPACK TPP вихідна матриця має розмірність 1000 * 1000. Перший рівень цього тесту не можна застосовувати в обчислювальних

системах, що дозволяють помістити всю вихідну матрицю в кеш-пам'яті. В цьому випадку отримані результати можуть істотно перевищувати реальні можливості системи. Застосування тестів LINPACK DP, LINPACK TPP для систем з масовим паралелізмом призводить до неадекватної оцінки їх продуктивності. Для оцінки продуктивності таких систем використовується тест LINPACK HPC (Highly Parallel Computing), який забезпечує повне завантаження обчислювальних ресурсів системи з масовим паралелізмом, збільшуючи розміри матриці. Варіант цього тесту розроблений і для паралельних обчислювальних систем. Тестовий пакет LINPACK з подвійною точністю широко використовується творцями високопродуктивних обчислювальних систем.

Тест LINPACK використовується при складанні рейтингу високопродуктивних комп'ютерів світу. За результатами тесту публікуються 500 найпродуктивніших обчислювальних комплексів. В даний час замість або поряд з бібліотекою LINPACK часто застосовується LAPACK, перероблена і адаптована під особливості сучасних обчислювальних машин бібліотека. LAPACK дозволяє вирішувати системи лінійних рівнянь, лінійні задачі найменших квадратів, знаходити значення і вектори матриць, а також сингулярні значення. Тест LAPACK спочатку був написаний на мові FORTRAN 77 і в даний час має велику кількість реалізацій від різних вендорів, доопрацювання яких дозволяють збільшити ефективність тестування.

2.3 Завдання дослідження

Як було описано вище в частині оцінки продуктивності ВПС було досягнуто значного прогресу, і в даний час існує декілька тестів, здатних об'єктивно оцінити продуктивність того чи іншого обчислювального комплексу, показати його ефективність і порівняти з іншими аналогами. Стосовно до систем віртуалізації робочих столів така методика не може бути застосована, так як потрібно підхід охоплює інші показники продуктивності.

Основною метою дослідження, описаного в даній дипломній роботі є розробка універсального методу порівняння протоколів доступу до ВРС, що дозволяє охопити найбільш затребувані ресурси, що відповідають за продуктивність. Прототипом до створення такого методу повинен стати тест, схожий ідейне з LINPACK тестом, задіює при своєму виконанні максимально повно ресурси ВПС.

Для реалізації методу необхідно провести детальний аналіз споживання основних обчислювальних ресурсів, більшою мірою впливають на роботу ВРС і на підставі даних аналізу буде запропонувати модель споживання ресурсів ВРС, таких як споживання ЦПП, ОЗП і ШПД, а також розроблений метод порівняння різних компонент продуктивності за допомогою застосуванням єдиного тимчасового критерію.

Критерії є транслятором величин продуктивності з різнорозмірних (вимірюваних в герцах, байтах, біт/секунду) в компоненти єдиного критерію, що виражається в однакових для всіх одиницях виміру - секундах (малюнок 2.2).

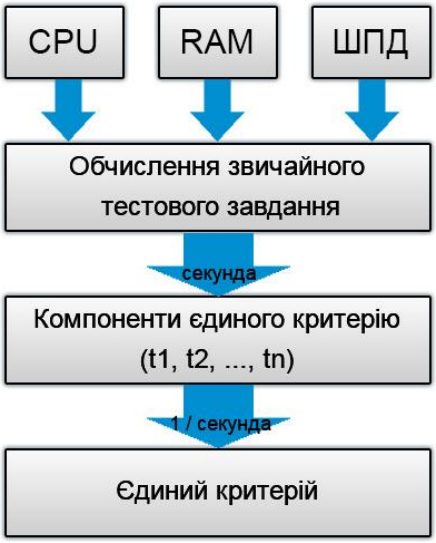


Рисунок 2.2 Перетворення в єдину розмірність

Метод передбачає використання складового високопродуктивного завдання, здатного розбити процес експерименту на окремі етапи, асоційовані з переважним споживанням певного ресурсу в конкретний момент.

Спираючись на отримані знання про якісне і кількісне споживання, в подальшому, дані переводяться в єдину шкалу за формою єдиного тимчасового критерію. Для автоматизації процесу отримання експериментальних даних, необхідних для порівняння продуктивності протоколів, потрібно створити ПЗ, яке працює відповідно до концепції узагальненого тимчасового критерію.

Програма для виконання тестування продуктивності VDI протоколів буде реалізована з урахуванням багатьох факторів, серед яких проста інтеграція в типову ОС користувача, що займається виконанням офісних завдань, установка без зміни реєстру ОС і точне вимірювання часових відрізків, що відповідають за кожен з етапів виконання тестового завдання.

2.3.1 Аналіз зв'язку між компонентами продуктивності

Сучасні ЕОМ будують відповідно до концепції фон Неймана. Відповідно до неї, ЕОМ в процесі обробки програми виконує циклічно певний набір кроків, ключовим кроком у концепції є безпосереднє виконання команди в ЦПП. Під час прорахунку завдання в ЦПП надходить

послідовність команд, якісь команди можуть бути виконані за 1 такт, інші за кілька. Сучасні обчислювальні процесори мають архітектуру ядра, що дозволяє використовувати кілька декодерів команд, а значить вони можуть завантажувати обчисленнями безліч обчислювальних блоків паралельно.

Якщо алгоритм дозволяє використовувати цю можливість, то ЦПП може задіяти кілька таких декодерів. В цьому випадку за один такт можна виконати більше однієї команди.

Позначимо час, що показує сумарну тривалість виконання всіх тактів, необхідних для вирішення задачі (програми) через величину Tгос, яка визначається як кількість тактів, що ділиться на тактову частоту процесора.

$$T_{proc} = \frac{C_{proc}}{f_{cpu}} \tag{2.1}$$

де T_{proc} - загальний час виконання всіх тактів під час вирішення завдання; C_{proc} – кількість тактів, необхідне для вирішення завдання; f_{cpu} – тактова частота процесора.

Величина T_{proc} пропорційна тимчасової складності розраховується алгоритмом, що представляє собою функцію від розміру вхідних даних, що дорівнює максимальній кількості елементарних операцій, необхідних алгоритму для вирішення конкретного завдання. Твердження про пропорційності випливає з того, що при зростанні обчислювальної складності задачі, пропорційно збільшується і загальна кількість тактів, необхідне для її прорахунку, а це в свою чергу відіб'ється на збільшенні тимчасової складності і T_{proc}.

При оцінці веб-кешування на серверах вчені встановили, що коефіцієнт α змінювався в межах 0,75 ± 0,05, в роботах [11, 12, 13] автори повідомляють, що з великою ймовірністю запити користувачів до документів на веб-серверах будуть повторюватися для документів, які були запитані недавно і розподіл часу між запитами описується статечним законом з показником α в межах 0,5 - 0,7.

В роботі [10] проводився узагальнений аналіз по популярності веб-серверів, в якому автори вказують на те, що коефіцієнт альфа, в залежності від специфіки завдань і звернень, лежить в діапазоні 0,6 - 1.

Багато дослідників відзначали, що ранговий розподіл запитів, оброблюваних ЦПП і ОЗП, за популярністю убуває статечним чином з номером рангу запиту і характеризується закономірністю «10/90» (коли перші 10% найбільш популярних запитів споживають 90% оперативної пам'яті).

З формули видно зв'язок між кількістю унікальних процесів і загальною кількістю процесів в результаті виконання завдання. Таким чином процесор буде витрачати час на обробку тільки унікальних запитів, в той час як частина результатів буде оброблена за допомогою пам'яті. Тобто можливо перейти від обробки всіх K процесів до обробки N процесів, тим самим скорочуючи кількість оброблюваних процесів з формули в 1 - α раз. При цьому загальний час виконання завдання зміниться.

$$T_{calc} = T_{proc}(1 - \alpha) + T_{mem} \quad (2.2)$$

де T_{calc} - загальний час виконання завдання, при використанні оперативної пам'яті, T_{proc} - час обчислення, при використанні тільки процесора, T_{mem} - час, необхідний для звернення до оперативної пам'яті.

Отриманий вираз показує зв'язок між процесами проходять в ЦПП і робочою в ОЗП. При постійній кількості операцій, необхідних для обчислення завдання, отримуємо залежність між тактовою частотою процесора і обсягом споживаної пам'яті.

Вираз може бути застосовано в разі, коли в пам'яті кешуються все унікальні запити, оброблювані в процесі обчислення, але при виконанні реальних завдань такого не відбувається і кешується лише частина запитів R_K (рис. 2.2).

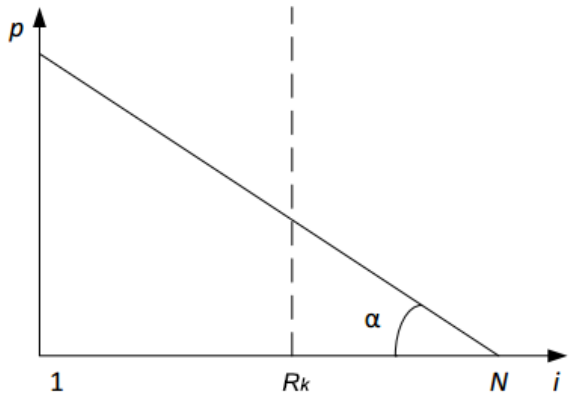


Рисунок 2.3 - Виділення кешованої частини запитів в ОЗП

У зв'язку з тим, що більшість сучасних завдань обчислюється розподілено і дані передаються по локальним, а іноді і глобальним обчислювальних мереж, то крім компонент T_{proc} і T_{mem} в загальний час виконання завдання також входить компонента T_{ping}, що відповідає за тимчасові затримки в передачі даних по каналах зв'язку.

Ця компонента залежить від ширини каналу передачі даних, ступеня його завантаженості, латентності при пересилання трафіку і варіації затримки на шляху від одного вузла мережі до

іншого. У підсумку загальний час обробки тестового завдання Ttest представлятиме є сумою трьох тимчасових компонент: компоненти обробки завдання в процесорі, ОЗУ і пересилання даних по мережі.

$$T_{test} = T_{proc} + T_{mem} + T_{ping} \quad (2.3)$$

Сучасні алгоритми обробки і передачі інформації в обчислювальних мережах націлені на зменшення компоненти Tping за рахунок зниження загальної кількості оброблюваної інформації і підвищення ефективності передачі даних в каналі зв'язку. Таким чином Tmem і Tping виявляються пов'язані між собою. Але ці залежності мають складний характер і в даний час недостатньо вивчені.

Для того щоб встановити характер взаємодії між величинами Tproc, Tmem і Tping в даній роботі пропонується використовувати підхід універсального тимчасового критерію, який повинен допомогти встановити взаємозв'язок між компонентами через загальну для всіх них величину - час.

2.4 Висновки до розділу 2

1. Розглянуті особливості роботи найбільш поширених VDI протоколів, серед яких протоколи VMware PCoIP і протоколи RDP двох реалізацій (VMware і Quest Software) на четвертому рівні моделі OSI. Особлива увага була приділена перевагам і недолікам застосування VDI, а також перераховані основні показники продуктивності VDI протоколів, що впливають на задоволеність користувача роботою за ВРС.

2. Проведено аналіз методів оцінки продуктивності хмарних обчислень, що використовуються для порівняння продуктивності різних обчислювальних комплексів. Розглядалися тести Whetstone, Dhrystone, LINPACK і LAPACK. На основі концепції однорозмірних уявлень, використовуваних в цих методах, був запропонований новий метод визначення продуктивності. Метод тимчасових інтервалів дозволяє врахувати внесок кожного з факторів продуктивності і дозволяє уявити величину кожного різнорозмірними фактора в єдиній для них усіх розмірності - розмірності часу, через рішення стандартної тестової задачі.

3. Виділені чинники, що впливають на продуктивність хмарних технологій, зокрема було приділено особливу увагу продуктивності процесора, оперативної пам'яті і впливу завантаженості каналу передачі даних на загальну продуктивність.

4. Запропоновано метод підвищення безпеки інформації, що пересилається по мережі між різними центрами обробки даних, за коштами додавання тимчасових міток в мережеві пакети.

РОЗДІЛ 3
ПРАКТИЧНА РОЗРОБКА ДОДАТКУ НА ОСНОВІ МЕТОДУ ЄДИНОГО КРИТЕРІЮ

В цій главі йдеться мова про практичну розробку додатку на основі методу єдиного критерію. Експериментальні дослідження було вирішено провести в таким тестових умовах, що повністю повторюють реальну інфраструктуру віртуальних робочих столів в хмарному середовищі.

Окремо описується створене для експерименту програмне забезпечення **VDIProtocolTest**, що реалізує вимірювання за методом єдиного тимчасового критерію.

3.1 Єдиний критерій

Незважаючи на загальні гідності і переваги VDI рішень над «товстими» для користувача інфраструктурами існує невизначеність в частині підбору підходящого, максимально ефективного VDI рішення, такого, щоб користувачі ВРС не відчували незручності при роботі в хмарної інфраструктурі і у них створювалося відчуття локальності всього процесу, це стосується зокрема швидкості відгуку на його дію і роботу з периферійними пристроями.

Також важливо, щоб в умовах обмеженості наданих обчислювальних ресурсів, віртуальні машини, виділені користувачам, споживали обчислювальні ресурси максимально ефективно. Зокрема, оптимізація споживання ресурсів, при однакових вихідних, можлива через вибір менш витратного VDI протоколу.

Продуктивність VDI протоколів оцінюється за споживанням ресурсів віртуальної інфраструктури, яка, в свою чергу, вимірюється трьома залежними величинами: продуктивністю ЦПП, ОЗП і ШПД. Такий підхід є обмеженим, тому що не надає можливості представляти один параметр через інший і визначати відносну важливість кожного, оскільки кожна з цих величин вимірюється власними, що не пов'язаними один з одним, величинами (Гц, Байти, біт/с). порівняння можливо тільки між величинами, що виражаються в одній розмірності.

З огляду на досвід оцінки продуктивності ВПС вже наявними методами, описаними у другому розділі, таких як LINPACK, LAPACK і ін. було вирішено також здійснити перехід від багатовимірних величин вимірювання до єдиної величині.

В даній дипломній роботі пропонується використовувати новий підхід єдиного тимчасового критерію. Єдиний критерій повинен представляти собою величину, яка об'єднує в собі три основних впливають на продуктивність VDI протоколів показників (ЦПП, ОЗП і мережу). Ця величина повинна описувати кожен з факторів в єдину для всіх шкалу, що дозволить

виділити вплив кожної зі складових продуктивності з загальної величини споживання ресурсів, оцінити її внесок і порівняти з іншими показниками.

На відміну від LINPACK, в якому вимірювання здійснюються в кількості FLOPS (кількість оброблених операцій з плаваючою крапкою в секунду), в якості розмірності для єдиної величини узагальненого критерію, представляє показники, що впливають на продуктивність VDI протоколів, обрана мережева розмірність біт/с. Модель використання єдиного тимчасового критерію передбачає розбиття високо продуктивної обчислювальної задачі, виконуваної під час сеансу термінальній сесії користувача, на кілька простих, таким чином, щоб кожен часовий інтервал яскраво характеризував роботу в даний момент конкретного компонента продуктивності.

Наприклад, повинні виділяти такі стадії, як процес завантаження даних на ВМ, при цьому передача інформації йде по мережі і є можливість створення навантаження високої інтенсивності саме на мережеву компоненту. Можливо виділити стадію виконання будь-якого роду обчислень безпосередньо всередині ВМ, тоді основними споживаними ресурсами буде ЦПП і ОЗП. При цьому величина споживання ресурсів буде варіюватися від VDI протоколу до VDI протоколу. Однак внесок кожної з стадій не є величиною однозначно визначається часом її виконання.

Для того, щоб отримати високу гнучкість і масштабованість, не втративши при цьому універсальності, час виконання кожної стадії завдання має свою вагу. Вагові коефіцієнти покликані збалансувати вплив різних компонент тимчасового критерію і врахувати всі деталі тимчасових розподілів.

Це необхідно тому деякі тимчасові компоненти можуть мати велику абсолютну величину, але мати незначні відмінності по часу виконання для різних протоколів, а значить низьку відносну цінність в критерії.

Загальна формула обчислення єдиного критерію має вигляд:

$$Q = \frac{1}{T} = \frac{1}{\sum_{i=1}^{n} k_i t_i} \tag{3.1}$$

3.2 Вибір завдання для тестування

Як було описано вище, для переходу до єдиної розмірності в роботі пропонується використовувати тимчасові відрізки виконання складового високопродуктивного завдання. Таким завданням було обрано завдання кодування «сирого» нестислого зображення через підключення з використанням VDI протоколу по алгоритму JPEG.

Завдання обробки можна розбити на кілька етапів:

– завантаження нестислого зображення на віртуальну машину;
– обробка нестислого зображення алгоритмами JPEG;
– висновок перетвореного зображення на монітор клієнта.

На першому етапі до USB порту клієнта приєднується flash накопичувач, на якому знаходиться файл з стислому вигляді зображенням. Використовуючи механізм перенаправлення периферійних пристроїв, накопичувач стає доступним всередині віртуальної машини і доступним в віртуальної ОС як локально підключений. У момент початку обробки даних файл з flash носія завантажується в ВМ, створюючи навантаження на мережеві ресурси.

На другому етапі відбувається безпосередня обробка зображення по алгоритму JPEG, під час цього навантажуються обчислювальні ресурси ВМ (ЦПП і ОЗП).

На третьому етапі, відбувається вивантаження матриці обробленого зображення в монітор клієнта. Так як обробка відео виводу виробляється VDI протоколом, то обсяг споживання обчислювальних і мережевих ресурсів в значній мірі залежить від інтенсивності оновлення зображення робочого столу. У момент відео виводу відбувається постійне оновлення зображення, тому що починає безперервно виводитися матриця обробленого зображення, а це, в свою чергу, призводить до стрибка в споживанні процесорних і мережевих ресурсів [2, 12].

Таким чином на всіх трьох етапах обробки завдання, отримані дані про споживання ресурсів демонструють різний характер задіяння ЦПП, ОЗП і дозволяють отримати більш детальне уявлення як про споживані ізольованого ресурсу (з явно вираженим пріоритетом споживання), так і при комбінованому використанні зрізу декількох. Час виконання кожної стадії експерименту представляється як компонент єдиного критерію.

За результатами експерименту визначається внесок кожної компоненти і призначається ваговий коефіцієнт, який включається в формулу розрахунку продуктивності методом єдиного тимчасового критерію. Величина єдиного критерію характеризує продуктивність VDI протоколу. У таблиці 3.1 показані компоненти єдиного критерію.

Таблиця 3.1 - Структура єдиного критерію

Позначення	**Опис**
T_0	Час завантаження завдання в віртуальну машину
T_1	Час виконання основних циклів VDIProtocolTest
T_2	Час виведення відеоданих в консоль клієнта
T_3	Загальний час операцій введення-виведення
T_Σ	Загальний час проведення експерименту

Продуктивність була виміряна для кожного протоколу, в залежності від розміру початкового нестислого зображення, яке визначається одним з наборів даних, представлених в таблиці 3.2.

Таблиця 3.2 - Набори вихідних даних

Набір даних	**Розмір зображення, кб**
S1	325
S3	645
S3	965
S4	1390
S5	1700

Критерій повинен бути універсальним, і його величина не повинна залежати від складності оброблюваної завдання та інфраструктури. Для того щоб задовольнити зазначеним умовам у формулі 3.1, яке описує концептуальне уявлення єдиного критерію, було вирішено використовувати розмір вихідної матриці в якості нормуючого множника. В такому разі формула 3.1 переходить в формулу 3.2, яка є більш універсальною, так як при збільшенні складності оброблюваного завдання (Збільшення розміру матриці нестислого зображення), крім збільшення загального часу T, відбувається збільшення і чисельника, а значить величина критерію в такому випадку залишається в рамках суворого діапазону. Показник критерію стає відносним, а не абсолютним.

$$Q = \frac{S}{T} = \frac{S}{\sum_{i=1}^{n} k_i t_i} \tag{3.2}$$

3.2.1 Запуск віртуальної машини Windows

Для проведення експериментів будемо застосовувати віртуальну машину Windows, за допомогою хмарного оточення Amazon AWS використовуючи шаблонний спосіб операційної системи. Застосування шаблону дозволяє розгортати віртуальні машини з повністю ідентичними налаштуваннями апаратної частини (ЦПП, ОЗП, жорсткі диски і ін.), А також ПЗ.

Ця властивість особливо важлива для експерименту, тому що реалізується принцип єдиних початкових умов. Для кожної ітерації експерименту створюється нова «чиста» віртуальна машина зі стандартними для експерименту налаштуваннями, без урахування впливу попередніх маніпуляцій. За допомогою Amazon EC2 можна вказати програмне забезпечення та параметри інстансу, який потрібно використовувати. На цьому екрані представлені варіанти вибору способу

машини Amazon (AMI), що представляє собою шаблон з необхідною для запуску інстансу конфігурацією програмного забезпечення. Я вибрав образ **Microsoft Windows Server 2012 R2 Base** і натиснув **Select**, як показано на малюнку 3.1.

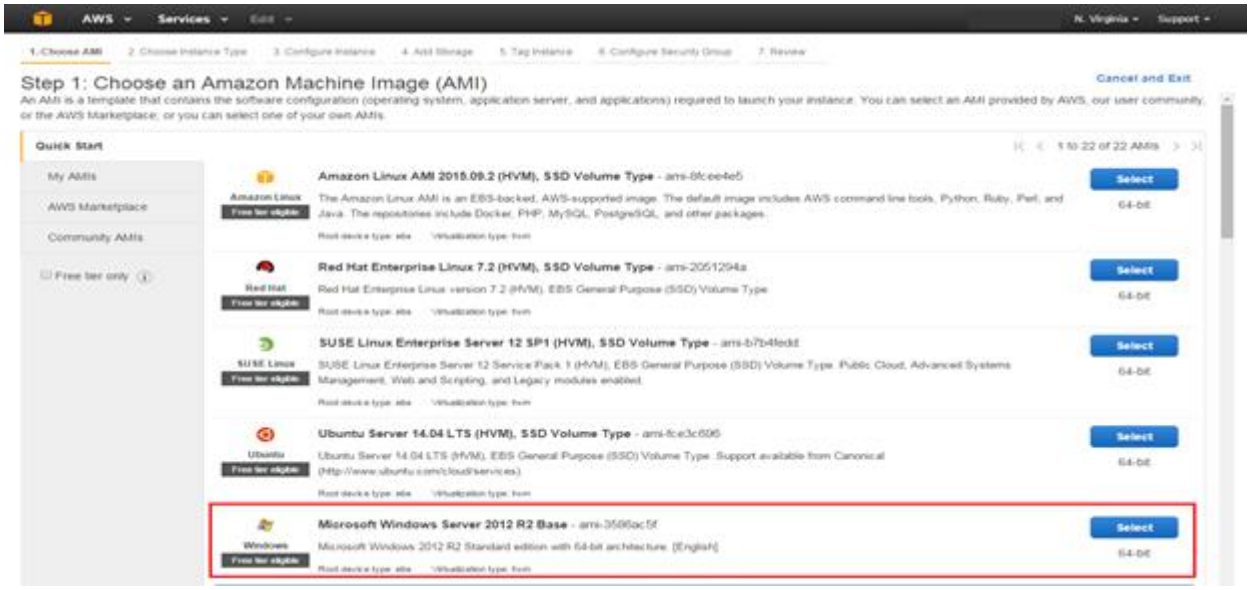


Рисунок 3.1 - Образ Microsoft Windows Server 2012 R2 Base

У даній роботі розглядалися дві реалізації RDP протоколу від компаній Quest Software і VMware, а також PCoIP протокол.

В Quest Software (Належить компанії Dell) використовується ліцензований Microsoft RDP протокол, покращений за допомогою графічного прискорювача EOP. RDP відноситься до TCP протоколів (з гарантованою доставкою пакетів), розроблений компанією Microsoft і спочатку вбудовується в операційні системи Windows.

Незважаючи на той факт, що RDP стандартизований, проте, реалізації від різних вендорів можуть демонструвати різну продуктивність за основними значимими параметрами. Крім RDP протоколу так само досліджувався протокол PCoIP - пропріетарний протокол, розроблений компанією Teradici і просувається компанією VMware.

PCoIP використовує UDP протокол, без гарантованої доставки пакетів. Крім різниці на транспортному рівні, ці протоколи використовують різні алгоритми кодування і стиснення даних, що передаються між сервером і клієнтом.

Доступ до клієнту здійснювався з використанням клієнта HP t5565 Thin Client. На клієнті встановлена операційна система HP Thin Pro. Це полегшена UNIX образна ОС зі скороченою кількістю функцій і програмного забезпечення, використовується тільки для локального входу на клієнт і запуску клієнтського ПЗ для доступу до віртуального робочого простору.

3.3 Програмне забезпечення для розрахунку єдиного критерію

Для проведення експерименту і виконання обчислення тимчасових відрізків і вирішення тестового завдання, було розроблено спеціалізоване ПЗ під назвою **VDIProtocolTest**, що реалізує алгоритм перетворення матриці нестислого зображення за засобами ДКТ (головний алгоритм JPEG-трансформації).

Основним завданням даного програмного забезпечення є завантаження ресурсів процесора і оперативної пам'яті віртуальної машини під час виконання ДКТ над нестислим зображенням, фіксація тимчасових інтервалів виконання всіх стадій тестового завдання і формування підсумкового звіту за результатами.

Крім даних про стадії роботи програми звіт також доповнюється даними від зовнішніх лічильників, що відповідають за відстеження показників завантаженості процесора, оперативної пам'яті і пропускної здатності мережі.

Структура вимірювального додатку представлена в наступному вигляді:

– прикладна задача: стиснення зображення за алгоритмом JPEG;
– вихідна функція: матриця нестислого зображення;
– основний алгоритм: Дискретне косинусе перетворення Фур'є;
– вихідна функція: матриця стисненого зображення;
– критерій оцінки: узагальнений критерій для оцінки продуктивності протоколів доступу до віртуального середовища;

Для того, щоб забезпечити просту інтеграцію **VDIProtocolTest** з віртуальними машинами користувачів, було прийнято рішення використовувати для написання додатку мову C# на платформі .NET, що дозволяє без установки додаткового ПЗ і зміни реєстру ОС інтегруватися з офісними додатками. Таким чином **VDIProtocolTest** може бути з легкістю запущений на будь-якій офісній машині. Крім цього, такий додаток не є оптимальним ПЗ для виконання JPEG перетворення, тому він буде в більшій мірі навантажувати обчислювальні ресурси ВМ, що в свою чергу в експериментальних умовах є плюсом.

Додаток запускає послідовність обробки, представлену на малюнку 3.2. Для виконання дискретної конусної трансформації потрібно два вкладених циклу, тіло циклів буде виконуватися $N \times N$ раз для кожного елемента матриці дискретної конусної трансформації.

Рисунок 3.2 - Алгоритм роботи програми

Для написання програмного забезпечення я використав мову програмування C# на платформі .NET, тому що на сьогоднішній момент мова програмування C# одна з найпотужніших, що швидко розвиваються і затребуваних мов в IT-галузі.

На даний момент на ньому пишуться найрізноманітніші програми: від невеликих десктопних додатків до великих веб-порталів і веб-сервісів, які обслуговують щодня мільйони користувачів.

У порівнянні з іншими мовами C# досить молодий, але в той же час він вже пройшов великий шлях. Перша версія мови вийшла разом з релізом Microsoft Visual Studio .NET в лютому 2002 року. Поточною версією мови є версія C# 7.0, яка вийшла в 7 березня 2017 року разом з Visual Studio 2017. C# є об'єктно-орієнтованим і в цьому плані багато перейняв у Java і C ++.

Наприклад, C# підтримує поліморфізм, успадкування, перевантаження операторів, статичну типізацію. Об'єктно-орієнтований підхід дозволяє вирішити завдання з побудови великих, але в той же час гнучких, масштабованих і розширюваних додатків. І C# продовжує активно розвиватися, і з кожною новою версією з'являється все більше цікавих функцій, як, наприклад, лямбда, динамічне зв'язування, асинхронні методи і т.д.

Фреймворк .NET представляє потужну платформу для створення додатків. Можна виділити наступні її основні риси:

– підтримка декількох мов. Основою платформи є загальномовне середовище виконання Common Language Runtime (CLR), завдяки чому .NET підтримує кілька мов: поряд з C# це також VB.NET, C++, F#, а також різні діалекти інших мов, прив'язані до .NET, наприклад, Delphi. NET. При компіляції коду на будь-якому з цих мов компілюється в збірку спільною мовою CIL (Common Intermediate Language) - свого роду асемблер платформи .NET. Тому ми можемо зробити окремі модулі однієї програми на окремих мовах;

– кросплатформеність. .NET є переносною платформою (з деякими обмеженнями). Наприклад, остання версія платформи на даний момент .NET Framework підтримується на більшості сучасних ОС Windows (Windows 10 / 8.1 / 8/7 / Vista). А завдяки проекту Mono можна створювати додатки, які будуть працювати і на інших ОС сімейства Linux, в тому числі на мобільних платформах Android і iOS;

– потужна бібліотека класів. .NET представляє єдину для всіх підтримуваних мов бібліотеку класів. І який би додаток ми не збиралися писати на C# - текстовий редактор, чат або складний веб-сайт - так чи інакше ми задіємо бібліотеку класів .NET;

– різноманітність технологій. Загальномовне середовище виконання CLR і базова бібліотека класів є основою для цілого стека технологій, які розробники можуть задіяти при побудові тих чи інших додатків. Наприклад, для роботи з базами даних в цьому стеку технологій призначена технологія ADO.NET. Для побудови графічних додатків з багатим насиченим інтерфейсом - технологія WPF. Для створення веб-сайтів - ASP.NET і т.д.

Як IDE для розробки програми я вибрав **Visual Studio 2015**. Це зручна і функціональна середовище програмування на мові C#.

Створюємо новий додаток **Windows Forms Applications** (малюнок 3.3) і додаємо необхідні елементи на форму, малюнок 3.4.

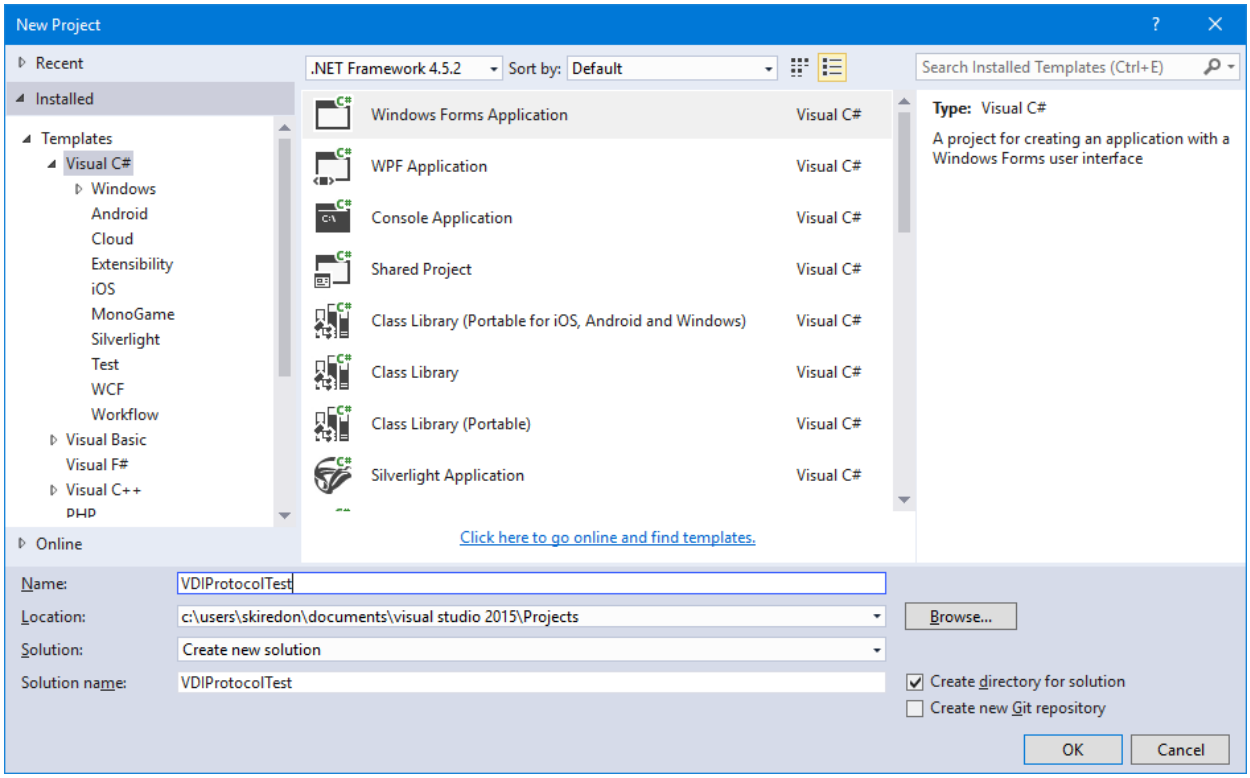


Рисунок 3.3 - Вибираємо Windows Forms Application

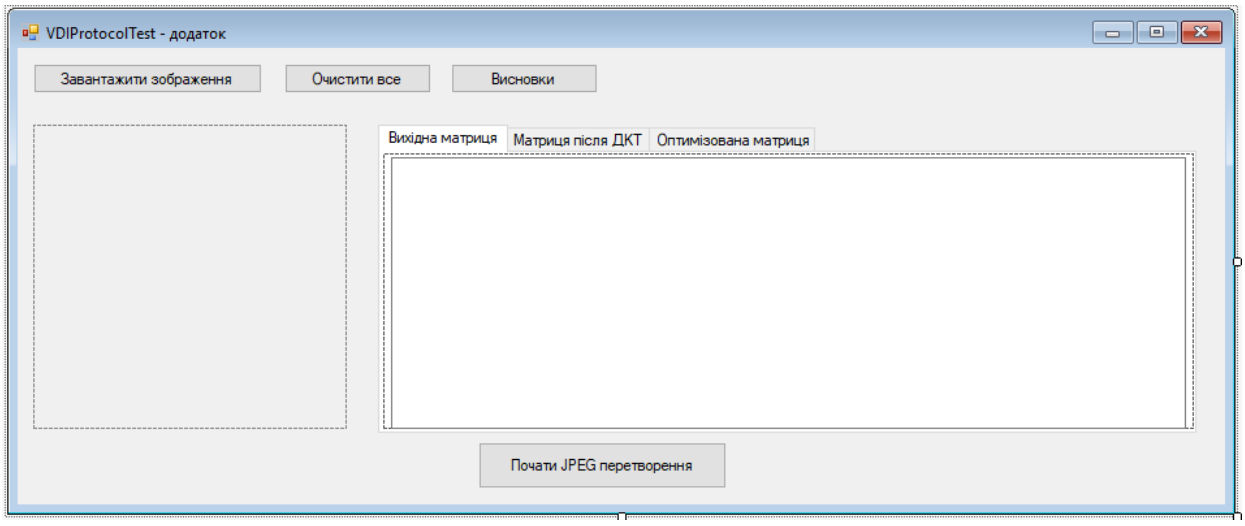


Рисунок 3.4 - Вікно програми **VDIProtocolTest**

Для виконання дискретної конусної трансформації потрібно створити нову функцію, і всередині неї застосувати два вкладених циклу. Увесь код показаний на малюнку 3.5.

```
private float[,] dctTransform(int[,] matrix)
{
    int m = 8;
    int n = 8;
    int i, j, k, l;
    float[,] dct = new float[m, n];
    float ci, cj, dct1, sum;

    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
        {
            if (i == 0)
                ci = 1 / (float)Math.Sqrt(m);
            else
                ci = (float)Math.Sqrt(2) / (float)Math.Sqrt(m);
            if (j == 0)
                cj = 1 / (float)Math.Sqrt(n);
            else
                cj = (float)Math.Sqrt(2) / (float)Math.Sqrt(n);

            sum = 0;
            for (k = 0; k < m; k++)
            {
                for (l = 0; l < n; l++)
                {
                    dct1 = matrix[k, l] *
                           (float)Math.Cos((2 * k + 1) * i * Math.PI / (2 * m)) *
                           (float)Math.Cos((2 * l + 1) * j * Math.PI / (2 * n));
                    sum = sum + dct1;
                }
            }
            dct[i, j] = (float)Decimal.Round((decimal)(ci * cj * sum), 2);
        }
    }
    return dct;
}
```

Рисунок 3.5 Код функції виконання ДКТ

У якості вхідних даних у функцію є двовимірний масив. Він зберігає в собі дані про зображенні, яке ми завантажуємо в програму і використовуємо для нашого тестування. Далі масив розбивається на блоки розміром 8 х 8 і застосовує до кожного алгоритм ДКТ.

VDIProtocolTest дозволяє ефективно завантажувати обчислювальні ресурси і мережу, а також забезпечує гнучке управлінням об'ємом вхідних і вихідний даних, за рахунок застосування різних розмірів матриць вхідного зображення з числа вже заготовлених або ж встановити необхідний розмір, роблячи можливим аналіз таких властивостей протоколів як швидкість і ступінь стиснення при передачі, оптимальність використання виділеної пам'яті і процесорного часу.

Інтерфейс програми реалізований у вигляді графічного вікна з можливістю введення і вибору вихідних даних, а також кнопок, призначених для виконання завдання і перегляду результатів. Всі органи управління інтерфейсом рознесені по трьох зонах: зона конфігурації, зона виконання і зона роботи з результатами перетворення.

Такий поділ дозволяє легко розібратися з управлінням програмою навіть користувачеві, який ніколи раніше не мав з нею справу, що дозволяє підключати до дослідженню більшу кількість людей - рядових користувачів, спрощуючи процес збору статистичних даних, задіючи безліч робочих станцій.

На малюнку 3.6 показана головна сторінка додатка **VDIProtocolTest**.

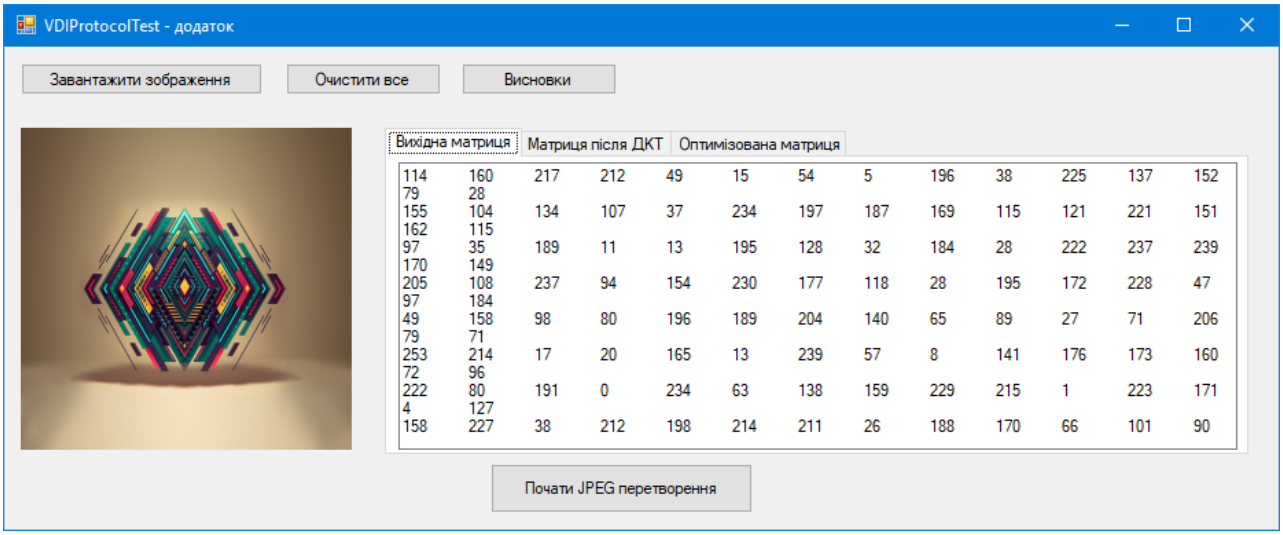


Рисунок 3.6 - Вікно програми в роботі

З головної вкладці запускається алгоритм JPEG-перетворення. Також є кнопка «Завантажити зображення», при натисканні якої можна вибрати інше зображення для тесту.

Опція розширеного відео виводу дозволяє провести перетворення ДКТ без виведення даних на монітор клієнта, такий режим дозволяє більш точно проаналізувати поведінку протоколів на стадії обчислень.

Також на сторінці присутні кнопки, що відповідають для відображення всіх проміжні результати: матриця нестислого зображення, матриця коефіцієнтів після ДКТ, оптимізована матриця після округлення і стиснення (малюнки 3.7, 3.8).

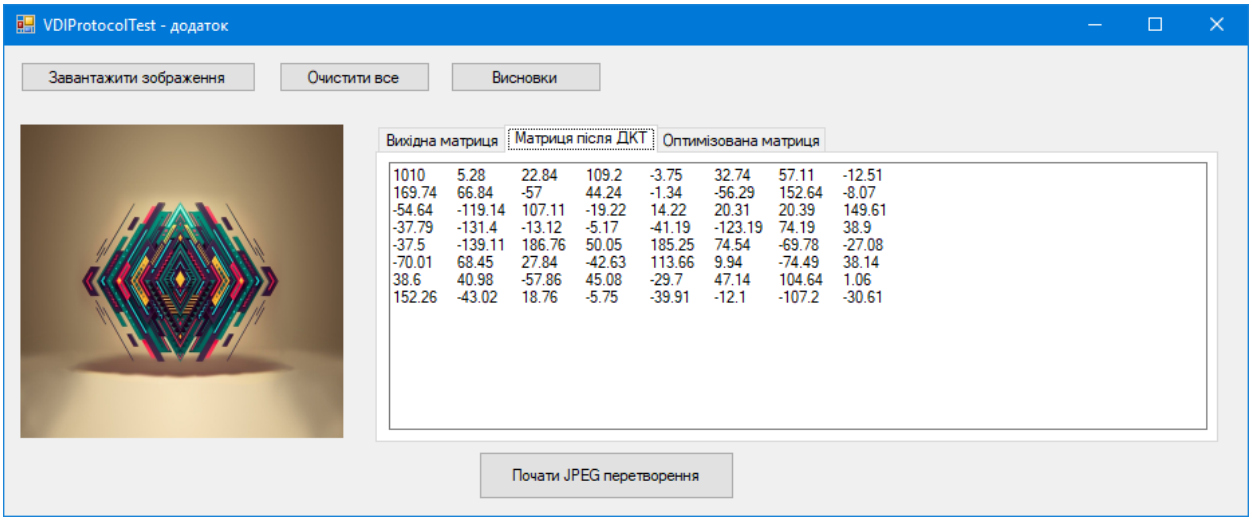


Рисунок 3.7 - Матриця коефіцієнтів після ДКТ

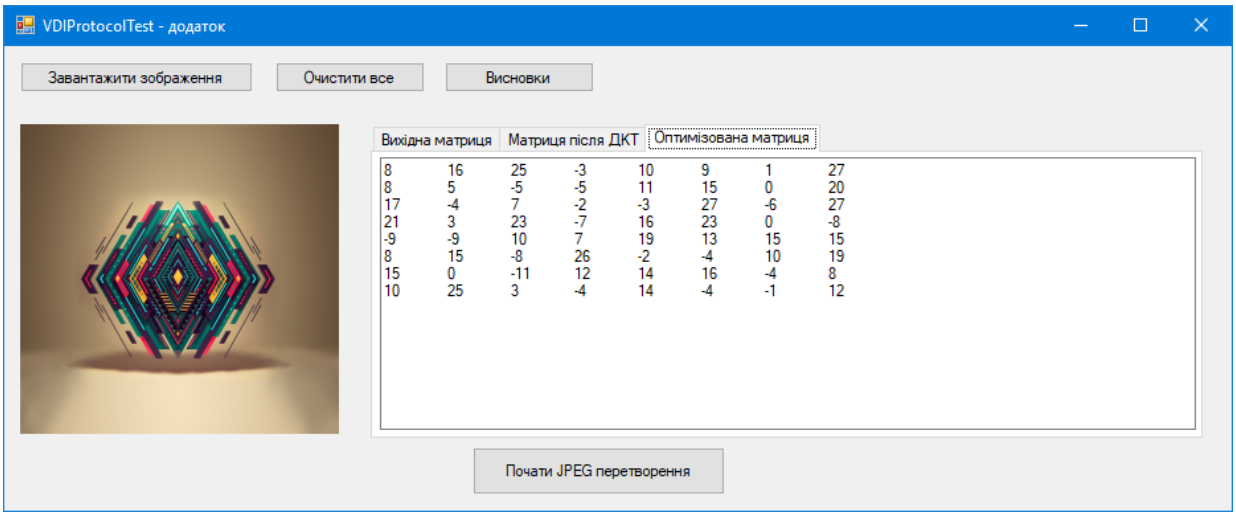


Рисунок 3.8 - Оптимізована матриця після округлення і стиснення

Кнопка «Висновки» відкриває сторінку, яка показує результати вимірювань: загальний час запуску коду **VDIProtocolTest**, час виконання основних циклів, час виконання операцій введення-виведення, ступінь стиснення і т.д.

Надалі, використовуючи експериментальну інформацію з внутрішніх лічильників **VDIProtocolTest**, і дані отримані з зовнішніх засобів вимірювання таких як PerfMon компанії Microsoft, будується загальна картина споживання ресурсів в кожної з стадій стандартної задачі.

У разі необхідності початку нового експерименту або повторенні зробленого, є можливість очистити отримані дані звіту і запустити нове тестування, для цього використовуються кнопки «Очистити все» і «Почати JPEG перетворення». Сторінка, що відображає результати вимірювань, представлена на малюнку 3.9.

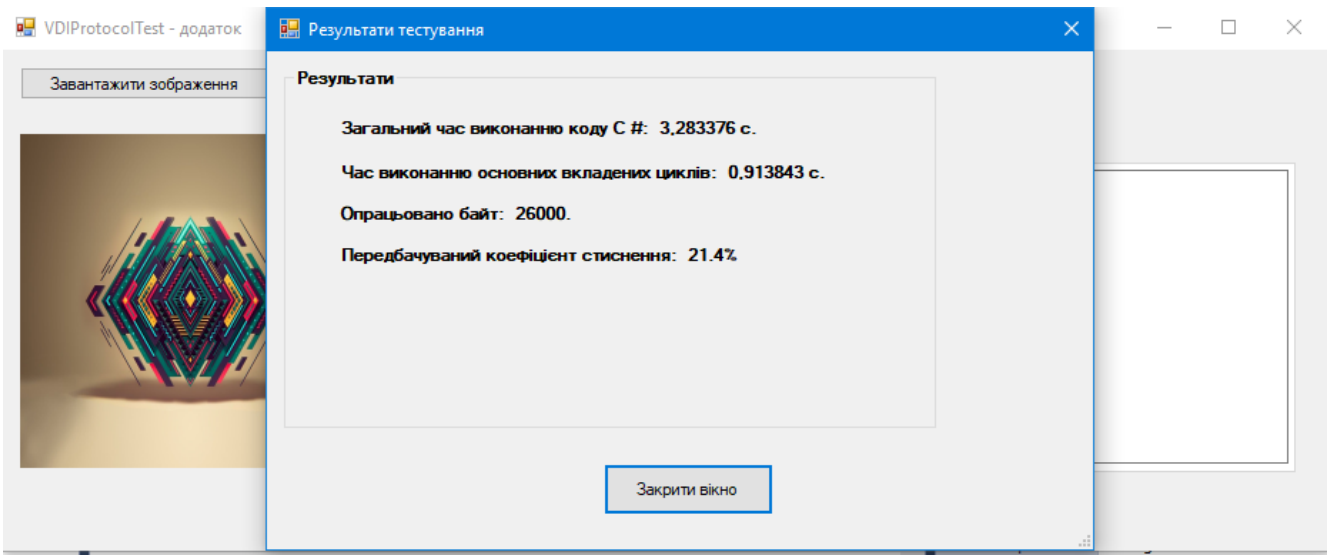


Рисунок 3.9 Вікно «Результати тестування»

Для проведення експериментальні досліджень була використана спеціально підготовлена інфраструктура віртуального робочого столу, описана в пункті 3.3. Під час експерименту основна взаємодія відбувається між клієнтом, що знаходиться на робочому місці користувача і віртуальною машиною, що знаходиться в «хмарі».

Крім встановленого ПЗ, необхідного для обробки графічних зображень, ВМ також оснащена засобами моніторингу, для отримання максимально повних даних про фактичне споживання ресурсів на всіх стадіях експерименту. Організаційна схема взаємодії клієнта і віртуальної машини, представлена на малюнку 3.10.

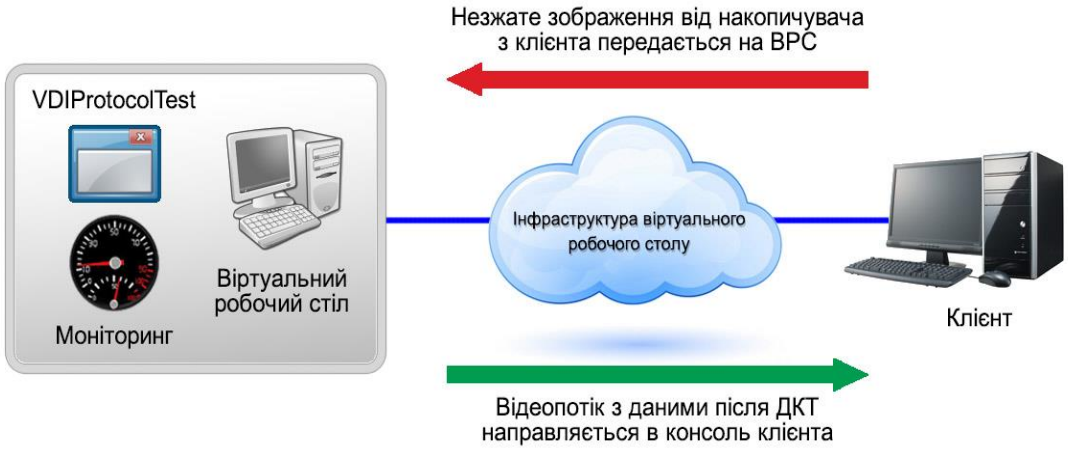


Рисунок 3.10 - Схема експериментального тестування

Експериментальні стадії можна розділити на зовнішні, що проходять без долі додатки **VDIProtocolTest** і внутрішні, що протікають безпосередньо в додатку.

Серед зовнішніх процесів можна виділити:

– встановлення VDI з'єднання і підключення до ВРС;
– передача нестислого зображення на ВРС через ЛВС;
– запуск зовнішніх програм моніторингу;
– збір даних з програм моніторингу;
– формування єдиного звіту за вимірюваннями;

Внутрішні процеси, що протікають в **VDIProtocolTest**:

– запуск внутрішніх лічильників для проміжних процесів;
– завантаження матриці нестислого зображення в додаток;
– виконання ДКТ над матрицею нестислого зображення;
– округлення коефіцієнтів матриці;
– стиснення матриці обробленого зображення;
– висновок на робочий стіл матриці після перетворень;
– збір даних з внутрішніх тимчасових лічильників;
– формування звіту на основі отриманих внутрішніх даних;

Алгоритм виконання внутрішніх і зовнішніх процедур представлений нижче на малюнку 3.11.

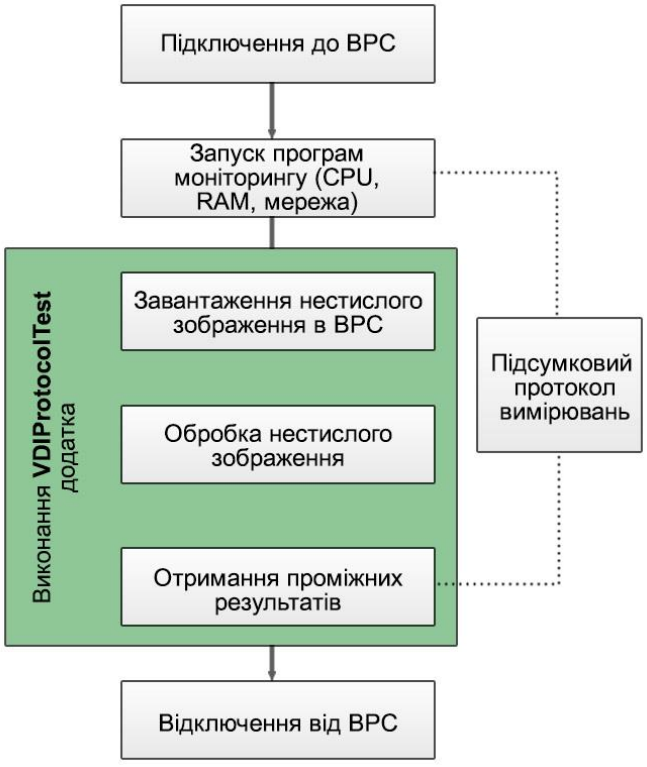


Рисунок 3.11 - Функціональна схема збору експериментальних даних

З боку VDI запускаються програми моніторингу, які фіксують зміни споживання ресурсів ВМ. Після старту програми **VDIProtocolTest**, зображення завантажується на ВРС і починається його JPEG-обробка (малюнок 3.11). Процес обробки зображення призводить до значної завантаженості системи.

Після закінчення експерименту додаток показує результати, готові для завантаження в протокол тестування. Параметри, які неможливо контролювати з боку **VDIProtocolTest**, вимірюються сторонніми програмними продуктами.

Під час виконання всіх циклів додатки фіксується завантаження ЦПП, ОЗП і швидкість передачі даних (ШПД). Експерименти з обробки матриці нестислого зображення проводилися по шість разів для кожного набору даних S1 ... S5.

Сценарії тестування розрізнялися як за обсягом матриці коефіцієнтів, так і по включеним опціям навантаження на обчислювальні ресурси хмари.

Далі в розділі наводяться дані експериментального дослідження, описаного в п'ятому розділі, а також проводиться аналіз компонентів єдиного тимчасового критерію, визначається, яка з компонентів єдиного критерію більшою мірою визначає продуктивність VDI протоколу і є домінантною, а які компоненти приблизно однакові для різних протоколів і їх вплив на продуктивність є незначним.

На завершення глави проводиться аналіз по продуктивності VDI протоколів RDP і PCoIP, дається порівняльна оцінка їх роботи.

3.3.1 Знаходження найбільш ефективного критерію

За результатами проведеного експерименту, була отримана інформація, що дозволяє наочно уявити структуру єдиного критерію у вигляді діаграм.

На діаграмах представлені дані за абсолютними значеннями окремих компонент тимчасового критерію, а також величини, наведені до відносної форми для отримання можливості оцінки їх частки в загальній величині критерію.

Всі результати досліджень отримані із застосуванням розробленого додатку **VDIProtocolTest**.

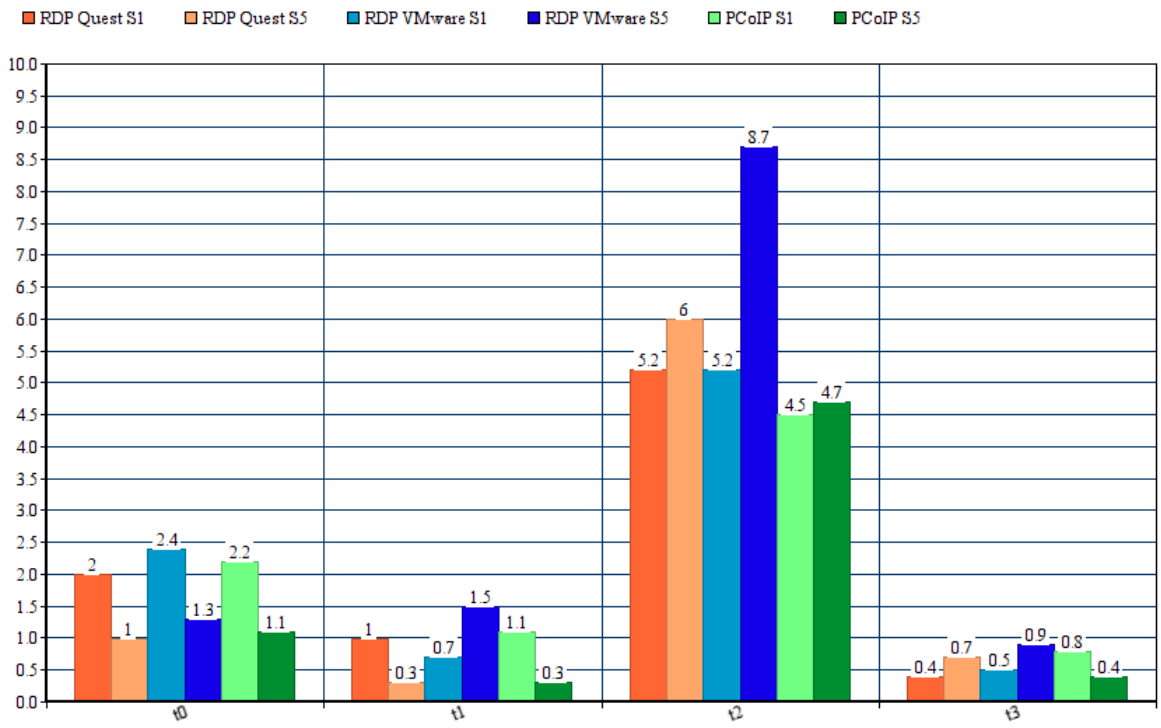


Рисунок 3.12 - Структура єдиного критерію для наборів даних S1 і S5

Для підвищення надійності отриманих даних, кожна контрольна точка була протестована 5 разів.

Малюнок 3.12 показує структуру узагальненого єдиного критерію при обробці різних розмірів матриць нестислого зображення протоколами PCoIP, RDP VMware і RDP Quest. Величини компонент, отримані експериментально представлені в нормованому вигляді, наведеному до розмірності часу обробки 1 КБ нестислого зображення протягом 1 секунди.

Аналізуючи структуру критерію, робимо висновок, що домінантною метрикою є компонент відповідає за час відео виводу зображення перетвореної матриці зображення (t2). Відносний час виконання компонент узагальненого критерію при використанні різних протоколів можна побачити на малюнку 3.9. згідно представлених даних видно, що інші компоненти узагальненого критерію мають приблизно однаковий час виконання на етапах тесту, не пов'язаних з відео висновком.

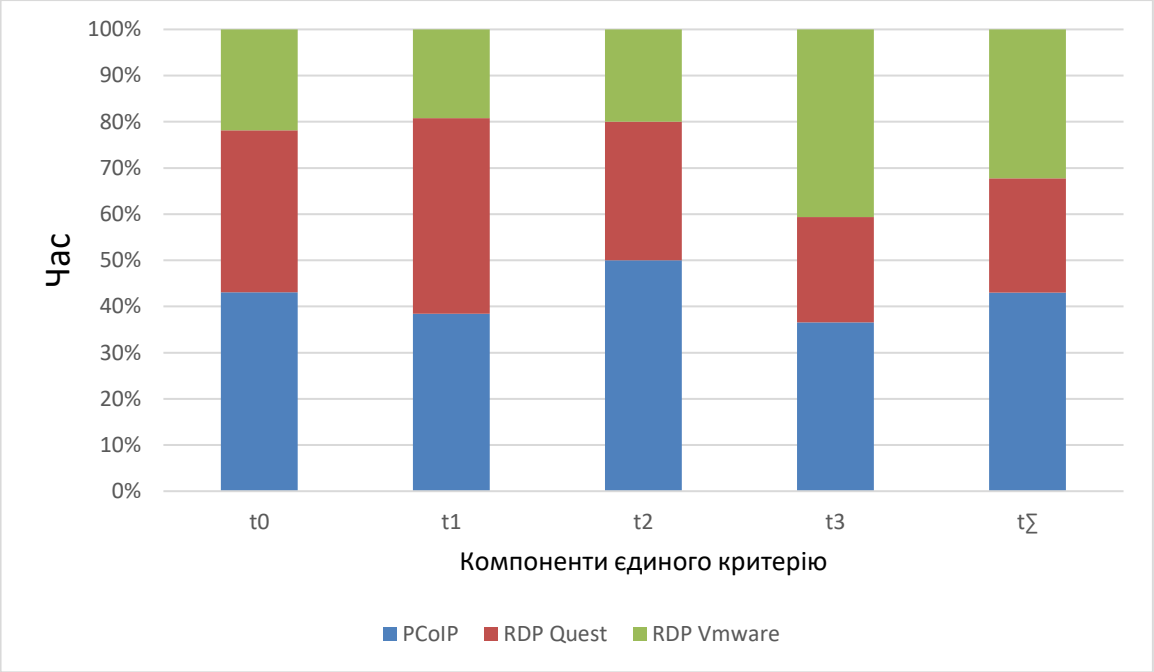
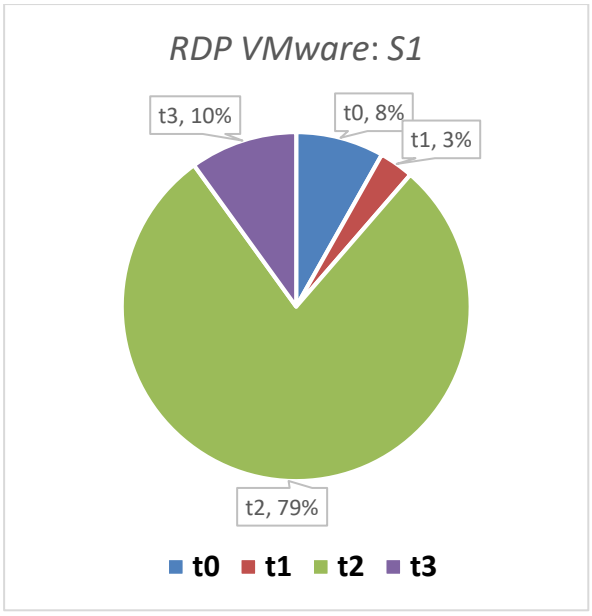


Рисунок 3.13 - Відносний час виконання компонентів єдиного критерію

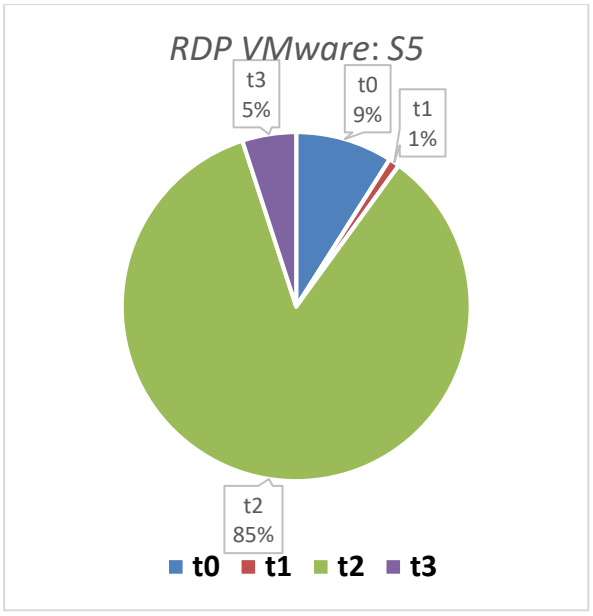
На малюнку 3.13 приведена структура єдиного критерію для протоколу PCoIP VMware і RDP VMware.

З кругових діаграм, представлених нижче видно, що при включеній опції розширеного відео виводу, для маленьких матриць (малюнки 3.14а і 3.14в) компонент t2 є домінуючим і впливає на загальний час тесту в більшій мірі.

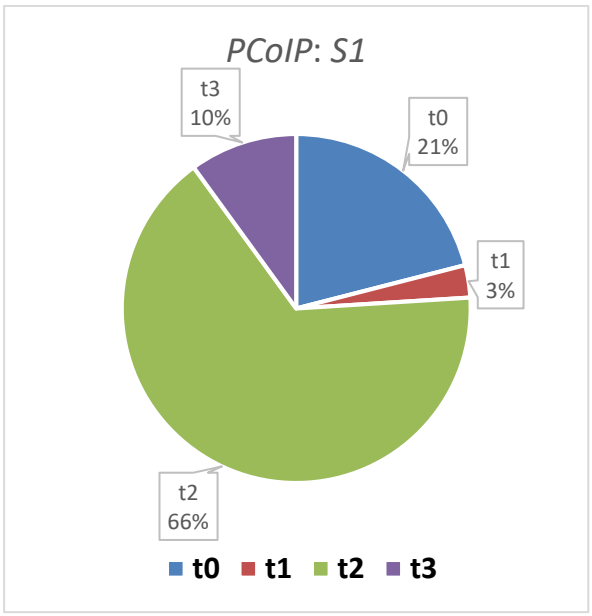
Для великих матриць відповідні компоненти мають схоже ставлення, тільки з більш вираженим пріоритетом t2 (малюнки 3.14б і 3.14г). Загальна частка, відведена на процес відео виведення, становить від 66% (при обробці малих матриць) і зростає з збільшенням складності оброблюваної завдання (85% для набору S5: RDP VMware). Час виконання основних циклів додатки (t1), яке описує ДКТ, практично однаково для всіх протоколів і не впливає на якість сервісу, що надається.



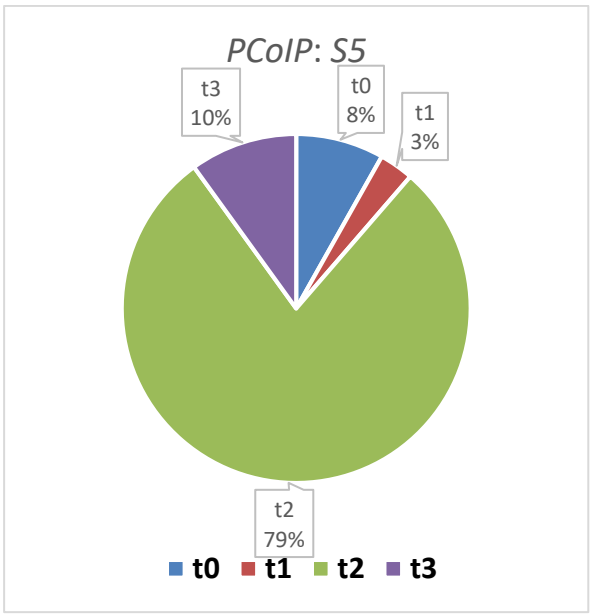
(а) *RDP VMware*: *S1*



(б) *RDP VMware*: *S5*



(в) *PCoIP*: *S1*



(г) *PCoIP*: *S5*

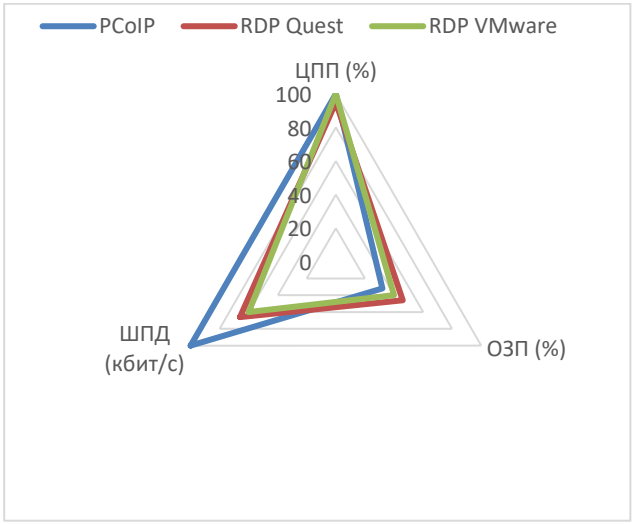
Рисунок 3.14 - Структура єдиного критерію

Згідно з результатами експериментів, основна значна різниця в продуктивності вноситься на стадії вивантаження даних в консоль тонкого клієнта.

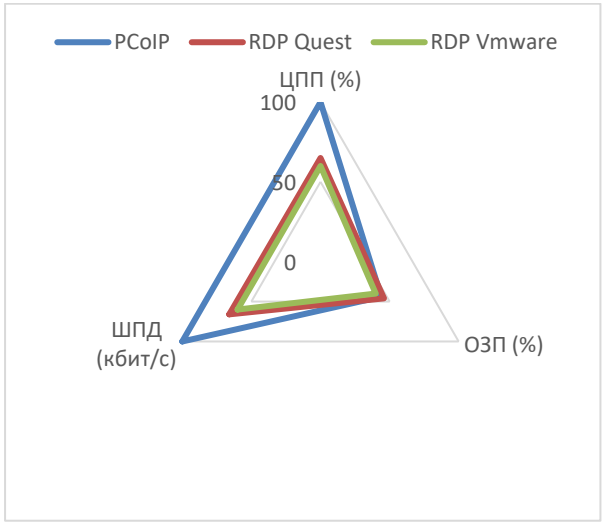
3.3.2 Дані про споживання обчислювальних і мережних ресурсів

Для того, щоб оцінити продуктивність VDI протоколів, зручно представити отримані дані у вигляді діаграм продуктивності (малюнок. 3.15). Діаграма продуктивності є 3D-діаграма, по

кожної з трьох осей якої, відкладається величина споживання ресурсів, вісь абсцис - відносне споживання ОЗП, вісь ординат - ЦПП, вісь аплікат - ШПД.



(а) Набір S1



(б) Набір S5

Рисунок 3.15 - Діаграми продуктивності для VDI протоколів

З наведених діаграм чітко видно, що такий ресурс, як ЦПП завантажується практично повністю (97 - 99%) тільки при використанні протоколу PCoIP.

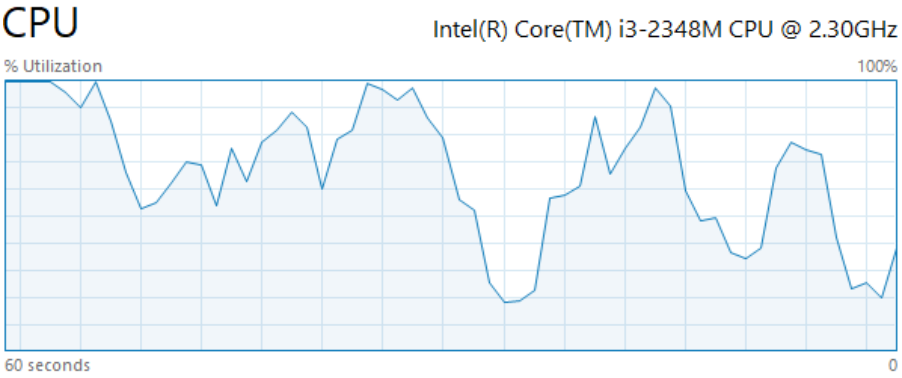
Протоколи RDP Quest і RDP VMware мають середнє інтегральне значення за цим показником 54% і 47% відповідно.

В протоколі RDP Quest використовуються оптимізовані кодеки, що сприяють прискоренню обробки графічних даних, що позначається на поліпшенні якості послуг, що надаються користувачеві, але ці кодеки не впливають на характеристики профілю продуктивності. Рисунок 3.16 демонструє характер залежності швидкості передачі даних по мережі від використовуваного VDI протоколу (представлені дані по протоколам RDP Quest і PCoIP VMware).

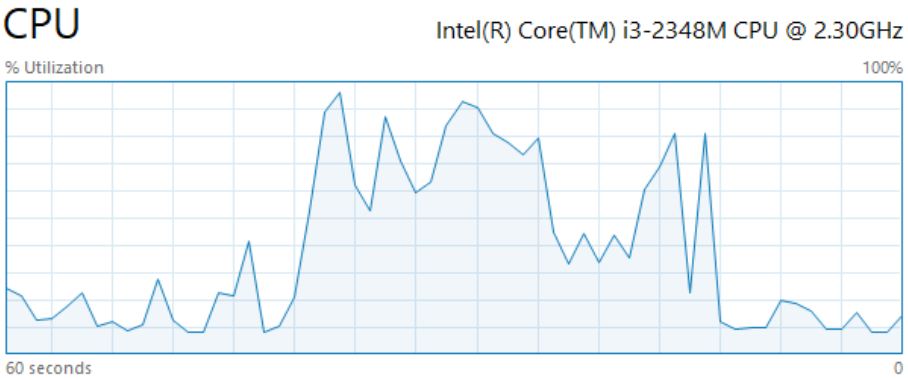
На представлених графіках ШПД чітко помітний первинний сплеск трафіку - це стадія завантаження зображення на ВМ. При цьому характер сплеску відрізняється в залежності від термінального протоколу. Для RDP він більш виражений і має велику амплітуду, для PCoIP більш згладжений.

Надалі починається стадія JPEG перетворення нестислого зображення і виведення потокового відео на монітор клієнта. Порівнюючи поведінку трафіку для цих етапів можна також помітити, що при використанні протоколу RDP, по-перше, вище пікові значення ШПД, а по-друге значення «скачуть»: є виражені провали і підйоми середнього значення ШПД.

PCoIP показує більш повне заповнення графіка, що говорить про те, що пропускна здатність мережі найбільш повно використовується протоколом PCoIP, який ефективно організовує передачу даних в прямому і зворотному напрямках, це особливо добре помітно при сильній завантаженості віртуальної машини.



(а) RDP Quest



(б) PCoIP

Рисунок 3.16 - ШПД і завантаження ЦПП для VDI протоколів під час виконання тесту

Слід також відзначити особливості роботи ЦПП. Говорячи про завантаження ЦПП з графіків видно, що під час передачі даних на клієнт, завантаження ЦПП у протоколів сімейства RDP має стрибкоподібний характер, а значить, з огляду на той факт, що завдання виконується довше, обчислювальні потужності простоюють.

Як показав експеримент середня величина завантаження процесора практично в 2 рази нижче, що проявляється в деколи недостатньою швидкістю обробки графічних процесів при роботі користувача з додатками, в яких основна ставка робиться на плавність і гладкість відпрацювання відеоефектів.

На відміну від RDP, ЦПП при використанні PCoIP завантажується рівномірно, направляючи свою обчислювальну потужність на відпрацювання графічних змін, це безумовно позитивним чином позначається на враженні користувача від роботи за ВРС.

Розглянемо продуктивність мережі для VDI протоколів залежно від розміру вихідної матриці нестислого зображення. Криві, представлені нижче, описують характер швидкості передачі даних в залежності від набору даних, що використовується при тестуванні, на малюнку 3.17.

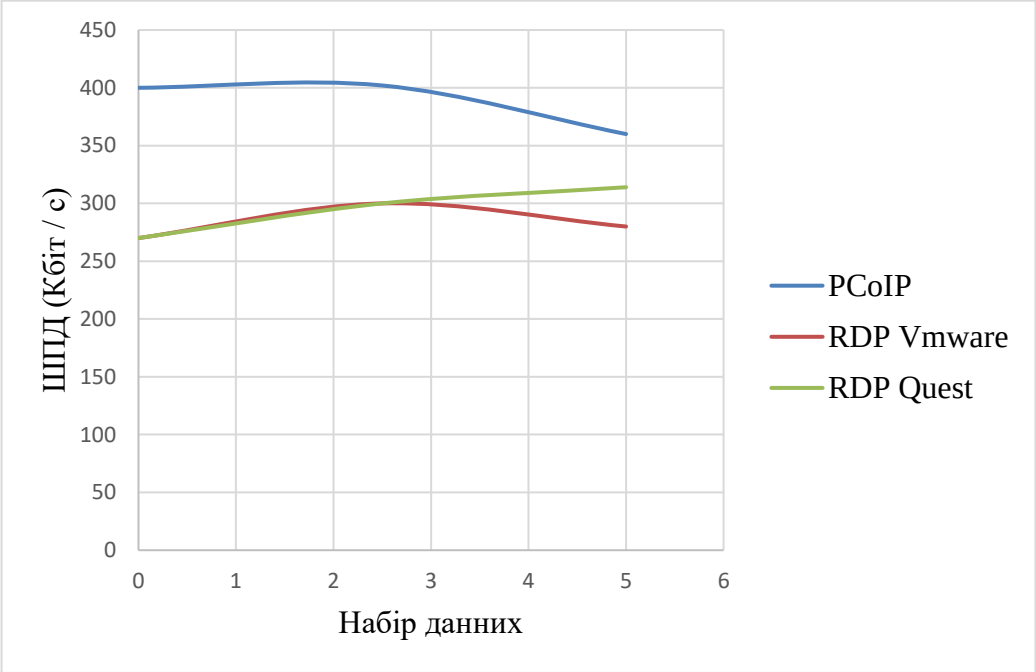


Рисунок 3.17 - Середнє значення ШПД мережі в залежності від розміру вихідної нестислого зображення

Тепер подивимося на дані по споживанню оперативної пам'яті. Малюнок 3.18 показує графіки, які містять залежності використання ОЗП від обсягу оброблених даних.

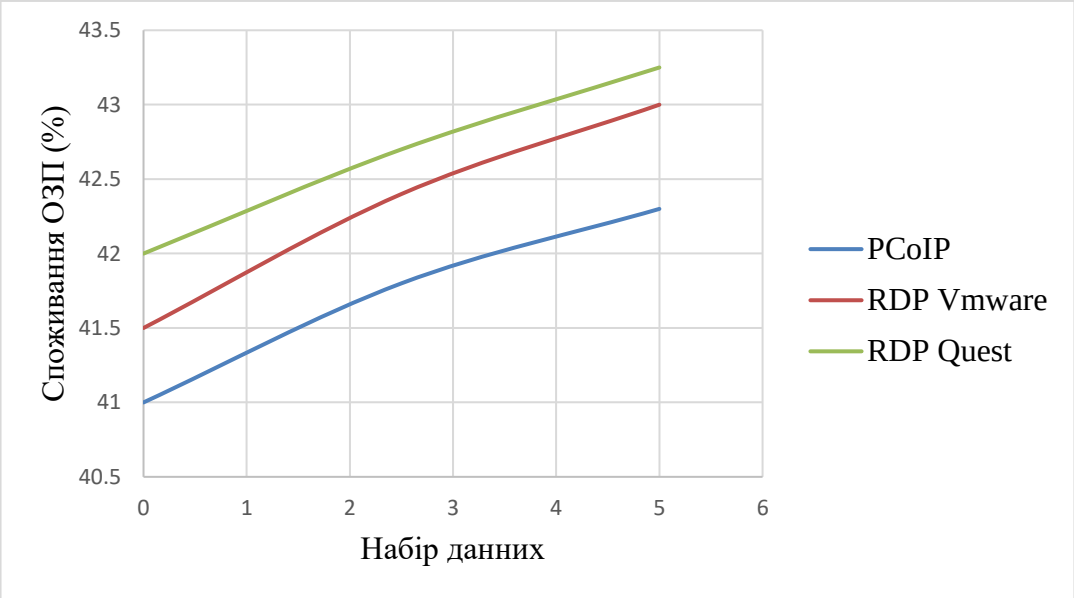


Рисунок 3.18 - Використання ОЗП (%)

Споживання оперативної пам'яті змінюється для всіх протоколів згідно лінійного закону розподілу.

Найкращі показники в частині споживання ОЗП демонструє протокол PCoIP з мінімальним рівнем споживання, гірше всіх показав себе RDP Quest.

Але з огляду на, що відмінність між гіршим і кращим результатом становить 1%, то різниця в споживанні цього ресурсу незначна.

Подальше збільшення завантаження ОЗП недоцільно, тому що при несуттєвому зміну приросту у використанні ОЗП, час обробки зображення збільшується значно.

3.4 Висновки до розділу 3

1. Запропоновано новий критерій оцінки продуктивності хмарних протоколів доступу до хмарного середовища з віртуальними робочими столами, згідно з яким розмірності всіх значущих величин, що беруть участь у формуванні продуктивності представляються у вигляді єдиної розмірності часу. Таке уявлення дає можливість наочно порівнювати на перший погляд не пов'язані показники, такі як споживання процесорних ресурсів, оперативної пам'яті і завантаження пропускної здатності каналу передачі даних в рамках єдиної осі.

2. Розроблена і обґрунтована стандартна тестова задача на основі JPEG перетворення, що дозволяє максимально різнобічно завантажити обчислювальні потужності віртуальної машини, що знаходиться в **хмарі** і отримати всебічні дані по продуктивності з розбивкою на окремі тимчасові інтервали. Особлива увага була приділена розробці програмного продукту, здатного в автоматичному режимі виконувати методику тестування і виводити після закінчення експерименту звіт з тимчасовими значеннями по кожному з етапів виконання тестової завдання. В результаті розроблене ПЗ вийшло універсальним в плані роботи з будь-якою версією операційної Windows і не вимагає установки додаткового програмного забезпечення, а також при запуску і роботі не модифікує налаштувань реєстру ОС;

3. Розглянуті питання, пов'язані з характером залежностей для графіків споживання мережевих ресурсів і, зокрема, рівномірність використання каналу передачі даних і обсяг переданих даних в залежності від використовуваного протоколу і розміру оброблюваного зображення;

4. Представлені, отримані в результаті експерименту, дані по споживанню обчислювальних і мережних ресурсів і проведено їх детальний аналіз. Зокрема, була встановлена основна метрика, яка є домінантним показником і в більшій мірі впливає на загальну продуктивність протоколів доступу до середовища хмарних обчислень. В якості такої метрики

виступила компонента, що відповідає за виведення відеоданих обробленого за алгоритмом JPEG зображення на монітор клієнта;

5. Отримана якісна оцінка протоколів доступу до середовища хмарних обчислень, яка свідчить про переваги використання протоколу PCoIP для доступу до віртуального робочого столу в хмарних середовищах. Використання такого протоколу дозволяє до 47% ефективніше використовувати ЦПП і до 25% пропускну здатність каналу передачі даних, при загальному зниженні кількості переданих даних, за рахунок більш ефективного стиснення потоку даних. Але підвищення ефективності використання ЦПП накладає додаткові вимоги до обчислювальних вимоги до використовуваних для підключення клієнтів;

6. Зроблено оцінку характеру споживання процесорних ресурсів, зроблені висновки про ефективність використання ЦПП для протоколів PCoIP і RDP.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ. ЕКОЛОГІЯ

В даному розділі був проведений аналіз шкідливих виробничих факторів, причин пожеж. Були розглянуті заходи, які дозволяють забезпечити виробничу санітарію та гігієну праці. Були розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики на підставі аналізу.

Виконується аналіз потенційно небезпечних і шкідливих виробничих чинників для персонального комп'ютера, на якому буде виконуватися розробка.

4.1 Загальні питання з охорони праці

В законі України «Про охорону праці» визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці.

4.1.1 Правові та організаційні основи охорони праці

Основним організаційним напрямом у здійсненні управління в сфері охорони праці є усвідомлення пріоритету безпеки праці і підвищення соціальної відповідальності держави, і особистої відповідальності працівників.

Наявні трудові відносини між працівниками і роботодавцями в Україні за темою дипломного проекту регулюються Кодексом законів про працю (КЗпП) України, відповідно до якого права працюючої людини на охорону праці охороняються всебічно та норми охорони праці неухильно інтегровані до правил внутрішнього розпорядку організації/підприємства.

4.1.2 Організаційно-технічні заходи з безпеки праці

В організації/підприємстві проводиться навчання і перевірка знань з питань охорони праці відповідно до вимог Типового положення про порядок проведення навчання і перевірки знань з питань охорони праці, затвердженого наказом Держнаглядохоронпраці України від 26.01.2005 N

15, зареєстрованого в Міністерстві юстиції України 15.02.2005 за N 231/10511 НПАОП 0.00-4.12-05 [1].

Також впроваджені організаційні заходи з пожежної безпеки – відповідно до вимог Типового положення про інструктажі, спеціальне навчання та перевірку знань з питань пожежної безпеки на підприємствах, в установах та організаціях України, затвердженого наказом Міністерства України з питань надзвичайних ситуацій та у справах захисту населення від наслідків Чорнобильської катастрофи від 29.09.2003 N 368, зареєстрованого в Міністерстві юстиції України 11.12.2003 за N 1148/8469 НАПБ Б.02.005-2003 [18].

Обов'язковими вимогами враховане наступне:

- не слід допускати до роботи осіб, що в установленому порядку не пройшли навчання, інструктаж та перевірку знань з охорони праці, пожежної безпеки;
- на підприємстві/організації, де експлуатуються ЕОМ з відео дисплейними терміналами (ВДТ) і периферійними пристроями (ПП), розробляється інструкція з охорони праці відповідно до Положення про розробку інструкцій з охорони праці [7];
- ознайомлення з правилами безпеки праці, засвідчується у журналі інструктажів;
- обов'язкові організаційні заходи перед початком, під час і після завершення роботи повинні включати перевірку (візуально) наявності і справності електрообладнання та його заземлення. Після закінчення роботи – вимагається прибирання робочого місця, відключення всіх електроприладів від електромережі.

4.2 Аналіз стану умов праці

4.2.1 Вимоги до приміщень

Геометричні розміри приміщення зазначені в табл. 4.1.

Таблиця 4.1 – Розміри приміщення

Найменування	**Значення**
Довжина, м	5
Ширина, м	5
Висота, м	3
Площа, $м^2$	25
Об'єм, $м^3$	75

Згідно з [2] розмір площі для одного робочого місця оператора персонального комп'ютера

має бути не менше 6 кв. м, а об’єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

Також робочий процес пов’язаний з багатьма документами, ля чого приміщення облаштований шафою для зручності. Задля дотримання визначеного рівня мікроклімату в будівлі встановлено систему опалення. Для забезпечення потрібного рівного освітленості кімната має вікно та систему загального рівномірного освітлення, що встановлена на стелі.

4.2.2 Вимоги до організації місця праці

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця за [3] (табл. 4.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 4.2 – Характеристики робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота робочої поверхні, мм	750	680 ÷800
Висота простору для ніг, мм	730	не менше 600
Ширина простору для ніг, мм	660	не менше 500
Глибина простору для ніг, мм	700	не менше 650
Висота поверхні сидіння, мм	470	400 ÷500
Ширина сидіння, мм	400	не менше 400
Глибина сидіння, мм	400	не менше 400
Висота поверхні спинки, мм	600	не менше 300
Ширина опорної поверхні спинки, мм	500	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	800	700 ÷800

Робочий стіл на досліджуваному місці також містить достатньо простору для ніг. Крісло, що використовується в якості робочого сидіння, є підйомно-поворотним, має підлокітники і можливість регулювання за висотою і кутом нахилу спинки, також воно м’яке і можливість

працювати у комфорті. Екран монітору знаходиться на відстані 0.8 м, клавіатура має можливість регулювання кута нахилу 5-15°. Отже, за всіма параметрами робоче місце відповідає нормативним вимогам.

Приміщення кабінету знаходиться на третьому поверсі трьох поверхової будівлі і має об'єм 78 $м^3$, площу – 25 $м^2$. У цьому кабінеті обладнано три місця праці, з яких два укомплектовані ПК.

Температура в приміщенні протягом року коливається у межах 18–24°С, відносна вологість — близько 50%. Швидкість руху повітря не перевищує 0,2 м/с. Шум в лабораторії знаходиться на рівні 50 дБА. Система вентилювання приміщення — природна неорганізована, а опалення — централізоване.

Розміщення вікон забезпечує природне освітлення з коефіцієнтом природного освітлення не менше 1,5%, а загальне штучне освітлення, яке здійснюється за допомогою восьми люмінесцентних ламп, забезпечує рівень освітленості не менше 200 Лк.

4.2.3 Навантаження та напруженість процесу праці

Під час виконання випускної роботи:

за фізичним навантаженням робота відноситься до категорії легкі роботи (Іа), її виконують сидячи з періодичним ходінням. Щодо характеру організування виконання дипломної роботи, то він підпадає під нав'язаний режим, оскільки певні розділи роботи необхідно виконати у встановлені конкретні терміни.

Рекомендовано застосування екранних фільтрів, локальних світлофільтрів (засобів індивідуального захисту очей) та інших засобів захисту, а також інші профілактичні заходи наведені в [3].

Роботу за дипломним проектом визнано, такою, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви – для розробників програм тривалістю 15 хв. Через кожну годину роботи;

4.3 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

4.3.1 Аналіз небезпечних та шкідливих факторів при виробництві (експлуатації) виробу

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 4.3). Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання НПАОП 0.00.-1.28-10 Правил охорони праці під час експлуатації електронно-обчислювальних машин», які встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з ВДТ і ПП. Основними робочими характеристиками персонального комп'ютера є:

- робоча напруга U=+220В +-5%;
- робочий струм I=2А;
- споживана потужність Р=350 Вт.

Таблиця 4.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
1	2	3	4
фізичні			
- підвищена температура поверхонь обладнання	експлуатація ЕОМ, принтерів, сканерів чи/або серверного обладнання для роботи	2	[2]
- підвищений рівень шуму на робочому місці	-//-	2	[4]
- підвищена або знижена вологість повітря	-//-	2	[2]
- підвищена або знижена рухливість повітря	-//-	1	[2]
- підвищений рівень іонізуючого випромінення в робочій зоні	-//-	2	[2] [15]
- підвищений рівень електромагнітного випромінення	-//-	2	[2]
- підвищений рівень напруги електричної мережі, замикання якої може відбутися через тіло людини	-//-	4	[16] [21]
- підвищений рівень статичної електрики	-//-	2	[16]
- підвищена напруженість магнітного поля	-//-	2	[16]
- недостатність природного світла	порушення умов праці (вимог до приміщень)	2	[20]

Продовження таблиці 4.3

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
1	2	3	4
- понижена контрастність	-//-	1	[3]
психофізіологічні:			
- нервово-психічна перевантаження (розумове, перенапруження аналізаторів-зорових)	- пошук інформації для постановки теми; - виконання роботи за темою диплома, тестування; - оформлення роботи	4	[17] [3]

Робочі місця мають відповідати вимогам Державних санітарних правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин, затверджених постановою Головного державного санітарного лікаря України від 10.12.98 N 7 [3].

4.3.2 Пожежна безпека

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється застосуванням розгалужених систем електроживлення ЕОМ, вентиляції і кондиціювання,. Небезпека загоряння пов'язана з особливістю комп'ютерів - із значною кількістю щільно розташованих на монтажній платі блоках електронних вузлів, схем, електричних і комутаційних кабелів, та ін. При відхиленні реальних умов експлуатації від розрахункових можуть виникнути пожежонебезпечні ситуації.

Висока щільність елементів в електронних схемах призводить до значного підвищення температури окремих вузлів (80…100℃). При проходженні електричного струму по провідниках і деталей виділяється тепло, що в умовах їх високої щільності може привести до перегріву, і може служити причиною запалювання ізоляційних матеріалів.

Для відводу теплоти від ЕОМ діє потужна система кондиціонування. Тому кисень, як окиснювач процесів горіння, є в будь-якій точці приміщень обчислювального центру.

Потенційними джерелами запалювання можуть бути:

а) іскри і дуги короткого замикання;

б) електрична іскра при замиканні і розмиканні ланцюгів;

в) перегріви від тривалого перевантаження;

г) наявність речовин, нагрітих вище за температуру самозаймання;

Причинами можливого загоряння і пожежі можуть бути:

а) конструктивні недоліки устаткування;

б) коротке замикання в електричних мережах;

Продуктами згорання, що виділяються на пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор і ін. При горінні пластмас, окрім звичних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідридні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол [19].

Найбільш ймовірна небезпека внаслідок перевантаження напруги, розрядки зарядів статичної електрики, пошкодження обладнання та електропроводки. Електростатичний розряд виникає під час тертя двох ізольованих матеріалів.

Для зниження займистості і здатності поширювати полум'я кабелі покривають вогнезахисними покриттями. Проектом передбачено прокладати проводку: приховано, під знімною підлогою розділяючи негорючими діафрагмами, в малодоступних місцях.

4.3.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три- провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

4.4 Гігієнічні вимоги до параметрів виробничого середовища

4.4.1 Мікроклімат

Мікроклімат робочих приміщень – це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини з’єднанням температури, вологості, швидкості переміщення повітря. Оптимальні значення для температури, відносної вологості й рухливості повітря для зазначеного робочого місця відповідають [2] і наведені в табл. 4.4:

Таблиця 4.4 – Норми мікроклімату робочої зони об’єкту

Період року	**Категорія робіт**	**Температура C^0**	**Відносна вологість %**	**Швидкість руху повітря, м/с**
Холодна	легка-1 а	22 – 24	40 – 60	0,1
Тепла	легка-1 а	23 – 25	40 – 60	0,1

Дане приміщення обладнане системами опалення, кондиціонування повітря або припливно-витяжною вентиляцією. У приміщенні на робочому місці забезпечуються оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря у відповідності до [2]. Рівні позитивних і негативних іонів у повітрі мають відповідати [2].

Контроль параметрів мікроклімату в холодний і теплий період року здійснюється не менше 3-х разів на зміну (на початку, середині, в кінці).

4.4.2 Освітлення

Світло є природною умовою існування людини. Воно впливає на стан вищих психічних функцій і фізіологічні процеси в організмі. Хороше освітлення діє тонізуюче, створює гарний настрій, покращує протікання основних процесів вищої нервової діяльності.

Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла в світильниках загального освітлення, оскільки люмінесцентні лампи мають високу потужність (80 Вт), тривалий термін служби (до 10000 годин), спектральний складом випромінюваного світла, близький до сонячного. При експлуатації ЕОМ виконується зорова робота IVв розряду точності (середня точність). При цьому нормована освітленість на робочому місці (Ен) рівна 200 лк. Джерелом природного освітлення є сонячне світло.

У приміщенні, де розташовані ЕОМ передбачається природне бічне освітлення, рівень якого відповідає [11]. Джерелом природного освітлення є сонячне світло. Регулярно повинен проводитися контроль освітленості, який підтверджує, що рівень освітленості задовольняє ДБН і для даного приміщення в світлий час доби достатньо природного освітлення.

Розрахунок освітлення.

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/9, в побутових – 1/10:

$$S_b = \left(\frac{1}{5} \div \frac{1}{10}\right) \cdot S_n, \tag{4.1}$$

де S_b – площа віконних прорізів, $м^2$;

S_n – площа підлоги, м^2.

$S_n = a \cdot b = 5 \cdot 5 = 25$ м^2,

$S = 1/9 \cdot 25 = 2,7$ м^2.

Приймаємо 2 вікна площею S=1,6 м^2 кожне.

Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні.

Розрахунок кількості світильників n виробляється по формулі (4.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M}, \tag{4.2}$$

де E – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, м^2; S = 25 м^2;

Z – поправочний коефіцієнт світильника (Z = 1,15 для ламп розжарювання та ДРЛ; Z = 1,1 для люмінесцентних ламп) приймаємо рівним 1,1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п. – 0,575

M – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм (для ЛБ-80).

Підставивши числові значення у формулу (4.2), отримуємо:

$$n = \frac{300 \cdot 25 \cdot 1,1 \cdot 1,5}{5400 \cdot 0,575 \cdot 2} \approx 2,0$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, які складаються з двох люмінесцентних ламп загальною потужністю 160 Вт, напругою – 220 В.

4.4.3 Шум та вібрація, електромагнітне випромінювання

Рівень шуму, що супроводжує роботу користувачів персональних комп’ютерів (зумовлений як роботою системних блоків, клавіатури, так і друкуванням на принтерах, а також зовнішніми чинниками), коливається у межах 50–65 дБА [4]. У залах опрацювання інформації та комп’ютерного набору рівні шуму не повинні перевищувати 65 дБА.

Віброізоляція можливо здійснювати за допомогою спеціальної прокладки під системний блок, який послаблює передачу вібрацій робочого столу. Вібрація на робочому місці в

приміщенні, що розглядається, відповідає нормам [4].

4.4.4 Вентилювання

У приміщенні, де знаходяться ЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції (вентиляційні шахти), тобто при V приміщення > 40 м3 на одного працюючого допускається природна вентиляція. Цей метод забезпечує приток потрібної кількості свіжого повітря, що визначається в СНіП.

Також має здійснюватися провітрювання приміщення, в залежності від погодних умов, тривалість повинна бути не менше 10 хв. Найкращий обмін повітря здійснюється при наскрізному провітрюванні.

4.5 Заходи з організації виробничого середовища та попередження виникнення надзвичайних ситуацій

Відповідно до санітарно-гігієнічних нормативів та правил експлуатації обладнання наводимо приклади деяких заходів безпеки.

1) Заходи безпеки під час експлуатації персонального комп'ютера та периферійних пристроїв передбачають:

- правильне організування місця праці та дотримання оптимальних режимів праці та відпочинку під час роботи з ПК;
- експлуатацію сертифікованого обладнання;
- дотримання заходів електробезпеки;
- забезпечення оптимальних параметрів мікроклімату;
- забезпечення раціонального освітлення місця праці (освітленість робочого місця не перевищувала 2/3 нормальної освітленості приміщення);
- облаштовуючи приміщення для роботи з ПК, потрібно передбачити припливно-витяжну вентиляцію або кондиціювання повітря.

2) Заходи безпеки під час експлуатації інших електричних приладів передбачають дотримання таких правил:

- постійно стежити за справним станом електромережі, розподільних щитків, вимикачів, штепсельних розеток, лампових патронів, а також мережевих кабелів живлення, за допомогою яких електроприлади під'єднають до електромережі;

- постійно стежити за справністю ізоляції електромережі та мережевих кабелів, не допускаючи їхньої експлуатації з пошкодженою ізоляцією;
- не тягнути за мережевий кабель, щоб витягти вилку з розетки;
- не закривати меблями, різноманітним інвентарем вимикачі, штепсельні розетки;
- не підключати одночасно декілька потужних електропристроїв до однієї розетки, що може викликати надмірне нагрівання провідників, руйнування їхньої ізоляції, розплавлення і загоряння полімерних матеріалів;

Вимоги безпеки при надзвичайних ситуаціях:

1) При раптовому припиненні подачі електричної енергії вимкнути всі пристрої ПК в такій послідовності: периферійні пристрої, ВДТ, системний блок, стабілізатор (або блок безперервного живлення). Витягнути вилки з розеток.

2) При замиканні, перевантаженні електричного струму на електричному обладнанні, внаслідок ураження грозової блискавки та ймовірної небезпеки ураженням електричним струмом, приймають наступне:

- попередження замикання здійснюється правильним вибором, монтажем експлуатації мереж;
- застосування захисту схем у вигляді швидкодіючих реле, а також вимикачів, плавких запобіжників.

Також застосовують різні **електричні захисні засоби від ураження струмом**:

а) Ізолюючі - ізолюють людини від струмоведучих або заземлених частин, а так-же від землі. Вони діляться на основні та додаткові.

б) Основні - володіють ізоляцією, здатної довго витримувати робоче напругу електроустановки і тому ними дозволяється стосуватися струмоведучих частин, знаходячи-трудящих під напругою.

в) Запобіжні - володіють ізоляцією нездатною витримати робоча напруга електроустановки, і тому вони не можуть самостійно захищати людину від ураження струмом під цим напругою.

Розрахунок захисного заземлення (забезпечення електробезпеки будівлі).

Згідно з класифікацією приміщень за ступенем небезпеки ураження електричним струмом [9], приміщення в якому проводяться всі роботи відноситься до першого класу (без підвищеної небезпеки). Коефіцієнт використання вертикальних заземлювачів $\eta_в$ в залежності від розміщення заземлювачів та їх кількості знаходиться в межах 0,4…0,99. Взаємну екрануючу дію горизонтального заземлювача (з'єднувальної смуги) враховують за допомогою коефіцієнта використання горизонтального заземлювача η_c.

Послідовність розрахунку.

1) Визначається необхідний опір штучних заземлювачів $R_{шт.з.}$:

$$R_{шт.з.} = \frac{R_д \cdot R_{пр.з.}}{R_{пр.з.} - R_д}, \quad (4.3)$$

де $R_{пр.з.}$ – опір природних заземлювачів; $R_д$ – допустимий опір заземлення. Якщо природні заземлювачі відсутні, то $R_{шт.з.} = R_д$.

Підставивши числові значення у формулу (А.3), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

2) Опір заземлення в значній мірі залежить від питомого опору ґрунту ρ, Ом·м. Приблизне значення питомого опору глини приймаємо ρ= 40 Ом•м (табличне значення).

3) Розрахунковий питомий опір ґрунту, $\rho_{розр}$, Ом·м, визначається відповідно для вертикальних заземлювачів $\rho_{розр.в}$, і горизонтальних $\rho_{розр.г}$, Ом·м за формулою:

$$\rho_{розр..} = \psi \cdot \rho, \quad (4.4)$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів I кліматичної зони з нормальною

вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в} = 1,7$ і горизонтальних $\rho_{розр.г} = 5,5$ Ом·м.

$$\rho_{розр.в} = 1,7 \cdot 40 = 68 \text{ Ом•м}$$

$$\rho_{розр.г} = 5,5 \cdot 40 = 220 \text{ Ом•м}$$

4) Розраховується опір розтікання струму вертикального заземлювача $R_в$, Ом, за (4.5).

$$R_в = \frac{\rho_{розр.в}}{2 \cdot \pi \cdot l_в} \cdot \left(\ln \frac{2 \cdot l_в}{d_{ст}} + \frac{1}{2} \cdot \ln \frac{4 \cdot t + l_в}{4 \cdot t - l_в} \right), \quad (4.5)$$

де $l_в$ – довжина вертикального заземлювача (для труб - 2−3 м; $l_в$=3 м);

$d_{ст}$ – діаметр стержня (для труб - 0,03−0,05 м; $d_{ст}$=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за ф. (4.6):

$$t = h_в + \frac{l_в}{2}, \quad (4.6)$$

де $h_в$ – глибина закладання вертикальних заземлювачів (0,9 м); тоді $t = 0,9 + \frac{3}{2} = 2,4$ м

$$R_{в} = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln \frac{2 \cdot 3}{0{,}05} + \frac{1}{2} \cdot \ln \frac{4 \cdot 2{,}4 + 3}{4 \cdot 2{,}4 - 3} \right) = 18.4 \text{ Ом}$$

5) Визначається теоретична кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання $\eta_в$:

$$n = \frac{2 \cdot R_{в}}{R_{д}} = \frac{2 \cdot 18{,}4}{4} = 9{,}2 \tag{4.7}$$

І визначається коефіцієнт використання вертикальних електродів групового заземлювача без врахування впливу з'єднувальної стрічки $\eta_в$ =0,57 (табличне значення).

6) Визначається необхідна кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$, шт:

$$n_{в} = \frac{2 \cdot R_{в}}{R_{д} \cdot \eta_{в}} = \frac{2 \cdot 18{,}4}{4 \cdot 0{,}57} = 16{,}14 \approx 16 \tag{4.8}$$

7) Визначається довжина з'єднувальної стрічки горизонтального заземлювача $l_с$, м:

$$l_{с} = 1{,}05 \cdot L_{в} \cdot (n_{в} - 1), \tag{4.9}$$

де $L_в$ – відстань між вертикальними заземлювачами, (прийняти за $L_в$ = 3м);

$n_в$ – необхідна кількість вертикальних заземлювачів.

$$l_{с} = 1{,}05 \cdot 3 \cdot (16 - 1) \approx 48 \text{ м}$$

8) Визначається опір розтіканню струму горизонтального заземлювача (з'єднувальної стрічки) $R_г$, Ом:

$$R_{г} = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot l_{с}} \cdot \ln \frac{2 \cdot l_{с}^{2}}{d_{см} \cdot h_{г}}, \tag{4.10}$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

$h_г$ – глибина закладання горизонтальних заземлювачів (0,45 м);

$l_с$ - довжина з'єднувальної стрічки горизонтального заземлювача $l_с$, м

$$R_{г} = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^{2}}{0{,}95 \cdot 0{,}15 \cdot 0{,}5} = 8{,}1 \text{ Ом}$$

9) Визначається коефіцієнт використання горизонтального заземлювача $\eta_с$. відповідно до необхідної кількості вертикальних заземлювачів $n_в$.

Коефіцієнт використання з'єднувальної смуги η_c=0,3 (табличне значення).

10) Розраховується результуючий опір заземлювального електроду з урахуванням з'єднувальної смуги:

$$R_{заг} = \frac{R_в \cdot R_г}{R_в \cdot \eta_c + R_г \cdot n_в \cdot \eta_в} \leq R_д. \quad (4.11)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: $R_{заг}$ <4 Ом, а саме:

$$R_{заг} = \frac{18,4 \cdot 8,1}{18,4 \cdot 0,3 + 8,1 \cdot 16 \cdot 0,57} = \frac{149,04}{79,392} = 1,87 \leq R_д$$

3) При виникненню пожеж при роботі на ПЕОМ від таких можливими джерел запалювання як:

- іскри і дуги коротких замикань;
- перегрів провідників, резисторів та інших радіодеталей ПЕОМ, від тривалої перевантаження та наявність перехідного опору;
- іскри при розмиканні і розмиканні ланцюгів;
- розряди статичної електрики;
- необережному поводженню з вогнем, а також вибухи газо-повітряних і паро-повітряних сумішей.

4.6 Охорона навколишнього природного середовища

4.6.1 Загальні дані з охорони навколишнього природного середовища

Діяльність за темою магістерської роботи, а саме: розробка програмного забезпечення в процесі її виконання впливає на навколишнє природнє середовище і регламентується нормами діючого законодавства.

Основним екологічним аспектом в процесі діяльності за даними спеціальностями є процеси впливу на атмосферне повітря та процеси поводження з відходами, які утворюються, збираються, розміщуються, передаються на віддалення (знешкодження), утилізацію, тощо в ІТ галузі.

В процесі діяльності виникають процеси поводження з відходами ІТ галузі. Нижче надано перелік відходів, що утворюються в процесі роботи:

- Відпрацьовані люмінесцентні лампи - І клас небезпеки;
- Батарейки та акумулятори (малі) - ІІІ клас небезпеки;

- Змінні носії інформації - IV клас небезпеки;
- Відходи друкуючих пристроїв - IV клас небезпеки;
- Макулатура - IV клас небезпеки;

Висновки до розділу 4

В цьому розділі був зроблений аналіз шкідливих та небезпечних чинників, з якими можна зіткнутися при роботі за комп'ютером.

Описано, що необхідно зробити для того, щоб приміщення відповідало необхідним нормам і було комфортним й безпечним для роботи, також було визначено параметри і певні характеристики приміщення.

Були приведені рекомендації щодо пожежної та електробезпеки, безпеки при надзвичайних ситуацій, організації робочого місця, наведені розміри приміщення та значення температури, вологості й рухливості повітря.

Також в роботі розрахував необхідну кількість і потужність ламп, захисне заземлення для забезпечення електробезпеки в будівлі та інших параметрів, значення яких впливає на умови праці, навів інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ ДО РОЗДІЛУ 4

1. НПАОП 0.00-4.12-05 Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці;
2. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень;
3. ДСанПіН 3.3.2.007-98 Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин;
4. ДСН 3.3.6.037-99 Санітарні норми виробничого шуму, ультразвуку та інфразвуку;
5. ДСТУ 4462.0.02:2005 Охорона природи. Комплекс стандартів у сфері поводження з відходами. Загальні вимоги;
6. ДБН В.2.5-56:2014 Системи протипожежного захисту;
7. НПАОП 0.00-4.15-98 Про розробку інструкцій з охорони праці;
8. НПАОП 0.00-6.03-93 Порядок опрацювання та затвердження власником нормативних актів про охорону праці;
9. НАПБ Б.03.002-2007 Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою;
10. НПАОП 40.1-1.01-97 Правила безопасной эксплуатации электроустановок;
11. НПАОП 40.1-1.32-01 Правила устройства электроустановок. Электрооборудование специальных установок;
12. ДСН 3.3.6.039-99 Санітарні норми виробничої загальної та локальної вібрації;
13. ДСТУ ГОСТ 12.1.012-90 ССБТ. Вібраційна безпека. Загальні вимоги;
14. ДБН В.2.5-67:2013 Опалення, вентиляція та кондиціонування;
15. ГОСТ 12.1.006-84 ССБТ. Електромагнітні поля радіочастот. Загальні вимоги безпеки. Допустимі рівні на робочих місцях і вимоги до проведення контролю;
16. ГОСТ 12.1.030-81 ССБТ. Електробезпечність. Захисне заземлення. Занулення;
17. НПАОП 0.00-1.28-10 Правила охорони праці під час експлуатації електронно-обчислювальних машин;
18. НАПБ Б.02.005-2003 Типове положення про інструктажі, спеціальне навчання та перевірку знань з питань пожежної безпеки на підприємствах, в установах та організаціях України;
19. ГОСТ 12.1.044-89 ССБТ. Пожежовибухонебезпека речовин і матеріалів. Номенклатура показників і методи їх визначення;
20. ДБН В.2.5-28:2015 Природнє і штучне освітлення;
21. ГОСТ 13109-97 «Електрична енергія. сумісність технічних електромагнітних. Норми якості електроенергопостачання загального призначення»;

ВИСНОВКИ

В ході науково - дослідницької діяльності введено основні поняття, пов'язані з технологіями хмарних обчислень, зокрема інфраструктурою віртуальних робочих столів і продуктивністю VDI протоколів.

Розглянуті особливості роботи найбільш поширених VDI протоколів, серед яких протоколи VMware PCoIP і протоколи RDP двох реалізацій (VMware і Quest Software) на четвертому рівні моделі OSI. Особлива увага була приділена перевагам і недолікам застосування VDI, а також перераховані основні показники продуктивності VDI протоколів, що впливають на задоволеність користувача роботою за ВРС.

Вирішено головне завдання: провести ґрунтовий аналіз сучасних принципів побудови хмарних ресурсів, систем віртуалізації. Розглянути особливості функціонування інфраструктури віртуальних робочих столів і протоколи доступу до середовища хмарних обчислень.

Сформовано мету роботи: підвищення ефективності використання протоколів доступу до середовища хмарних ресурсів шляхом модульного тестування і програмна реалізація вимірювального інструмента.

Створено універсальний тест продуктивності протоколів доступу до середовищі хмарних обчислень, що дозволяє порівнювати споживання обчислювальних ресурсів хмарної інфраструктури при підключенні до неї через протоколи віддаленого доступу.

Розроблено метод отримання даних для єдиного критерію, заснований на процесах передачі нестислого зображення по мережі на віртуальну машину, його перетворенні за алгоритмом JPEG всередині програмного забезпечення VDIProtocolTest і подальшої передачі вже обробленого зображення на монітор користувача.

Розроблено програмний вимірювальний інструмент VDIProtocolTest, що дозволяє автоматизувати процес збору експериментальних даних. VDIProtocolTest зроблений таким чином, що для офісного робочого місця не вимагає установки додаткових сторонніх програмних продуктів і не змінює реєстру операційної системи при установці і роботі.

Зроблено аналіз зв'язку часової і просторової обчислювальної складності при обробці завдань в обчислювальних системах. Для аналізу використовувався статистичний розподіл і оптимізація шляхом резервування запитів, переданих в оперативну пам'ять.

ПЕРЕЛІК ПОСИЛАНЬ

1. **Дроздова И. И., Жилин В. В.** Безопасность облачных хранилищ;
2. **Лось А.Б., Царегородцев А.В., Сорокин А.В.** Комплексный подход к построению защищенных информационно - телекоммуникационных систем на базе гибридной облачной среды // Национальная безопасность / nota bene. - 2016. - №3. - С. 332 - 337. DOI: 10.7256/2073-8560.2016.3.16542;
3. **Парфенов Ю.П., Девятериков Д.А.** Масштабируемое и отказоустойчивое dbass хранилище из линейки продуктов POSTGESQL// Программные системы и вычислительные методы. – 2014. – № 1. – С. 125 - 130. DOI: 10.7256/2305-6061.2014.1.10549;
4. **Царегородцев А.В., Ермошкин Г.Н.** Модель оценки рисков информационной безопасности информационных систем на основе облачных вычислений// Национальная безопасность / nota bene. – 2013. – № 6. – С. 46 - 54. DOI: 10.7256/2073-8560.2013.6.9585;
5. **Широкова Е.А.** Облачные технологии // Современные тенденции технических наук: матер. Междунар. науч. конф. (г. Уфа, октябрь 2011 г.). Уфа: Лето, 2011. С. 30—33;
6. **Ahmed, N.** Discrete cosine transform / N. Ahmed, T. Natarajan, K. R. Rao // Computers, IEEE Transactions. – 1974. – Vol. 100 № 1. – P. 90-93;
7. **Deboosere, L.** Efficient resource management for virtual desktop cloud computing / L. Deboosere, B. Vankeirsbilck, P. Simoens, F. De Turck, B. Dhoedt, P. Demeester // The Journal of Supercomputing. – 2012. – Vol. 62, № 2. – P. 741 -767;
8. **Dongarra, J. J.** The LINPACK benchmark: An explanation / J. J. Dongarra // Supercomputing. Springer Berlin Heidelberg. – 1988 – P. 456-474;
9. **Khmelevsky, Y.** Cloud computing infrastructure prototype for university education and / Y. Khmelevsky, V. Voytenko // Proceedings of the 15th Western Canadian Conference on Computing Education ACM. – 2010. – P. 8;
10. **Liao, X.** Towards virtualized desktop environment. / X. Liao, H. Jin, L. Hu, H. Liu // Concurrency and Computation: Practice and Experience. – Vol. 22 № 4. – P. 419-440;
11. **Luo, J. Z.** Cloud computing: architecture and key technologies / J. Z. Luo, J. H. JIN, A. B. SONG, F. Dong // Journal of China Institute of Communications. – Vol. 32, № 7. – P. 3-21.
12. **Michael J. Kavis.** Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS);
13. **Tim Mather, Subra Kumaraswamy, Shahed Latif.** Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance (Theory in Practice);

Електронні ресурси:

14. **Горбенко А.В., Тарасюк О.М., Лысенко А.С.** Анализ производительности web сервисов в среде Cloud Computing; (http://www.hups.mil.gov.ua/periodic app/article/3274/zhups_2013_2_22.pdf)
15. **Лозовюк А.** XML-RPC: вызов процедур посредством XML (http://citforum.ru/internet/xml/xml_rpc/);
16. Основы Облачных вычислений (по рекомендациям NIST) http://cloud.sorlik.ru/synopsis-4.html;
17. **Хабрахабр.** REST vs SOAP. Часть 1. Почувствуйте разницу (http://habrahabr.ru/post/131343);
18. ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing Internet Computing, September/October 2008 (vol. 12 no. 5), pp. 96-99 **Carl Hewitt**, Massachusetts Institute of Technology. URL: http://www.computer.org/csdl/mags/ic/2008/05/mic2008050096-abs.html;
19. **Sean Bianco,** RDP Vs HDX – ICA – Which one is better for your remote networks? (https://www.parallels.com/blogs/ras/ica);
20. **M. Sandbu**, Бенчмаркінг віддалених протоколів, Citrix, VMware та RDP-Part One PCoIP проти Blast Extreme; (https://msandbu.wordpress.com/2016/03/29/remote-protocols-benchmarking-citrix-vmware-and-rdppart-one-pcoip-vs-blast-extreme)
21. https://ru.wikipedia.org;
22. **W3.org.** Официальный сайт Консорциума Всемирной Паутины. Спецификация SOAP версия 1.2. (http://www.w3.org/2002/07/soaptranslation/russian/part0.html);

Додаток А

Вихідний код програмного забезпечення

Файл AssemblyInfo.cs

```
using System.Reflection;
using System.Runtime.CompilerServices;
using System.Runtime.InteropServices;

[assembly: AssemblyTitle("VDIProtocolTest")]
[assembly: AssemblyDescription("")]
[assembly: AssemblyConfiguration("")]
[assembly: AssemblyCompany("")]
[assembly: AssemblyProduct("VDIProtocolTest")]
[assembly: AssemblyCopyright("Copyright ©  2019")]
[assembly: AssemblyTrademark("")]
[assembly: AssemblyCulture("")]
[assembly: ComVisible(false)]
[assembly: Guid("21d539d9-0de9-401a-86b0-763041bb5eb7")]
[assembly: AssemblyVersion("1.0.0.0")]
[assembly: AssemblyFileVersion("1.0.0.0")]
```

Файл Form1.Designer.cs

```
namespace VDIProtocolTest
{
   partial class Form1
   {

      private System.ComponentModel.IContainer components = null;

      protected override void Dispose(bool disposing)
      {
         if (disposing && (components != null))
         {
            components.Dispose();
         }
         base.Dispose(disposing);
      }

      #region Windows Form Designer generated code

      private void InitializeComponent()
      {
         this.button1 = new System.Windows.Forms.Button();
         this.button2 = new System.Windows.Forms.Button();
```

```
this.button3 = new System.Windows.Forms.Button();
this.openFileDialog1 = new System.Windows.Forms.OpenFileDialog();
this.pictureBox1 = new System.Windows.Forms.PictureBox();
this.textBox1 = new System.Windows.Forms.TextBox();
this.tabControl1 = new System.Windows.Forms.TabControl();
this.tabPage1 = new System.Windows.Forms.TabPage();
this.tabPage2 = new System.Windows.Forms.TabPage();
this.textBox2 = new System.Windows.Forms.TextBox();
this.tabPage3 = new System.Windows.Forms.TabPage();
this.textBox3 = new System.Windows.Forms.TextBox();
this.button4 = new System.Windows.Forms.Button();
((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
this.tabControl1.SuspendLayout();
this.tabPage1.SuspendLayout();
this.tabPage2.SuspendLayout();
this.tabPage3.SuspendLayout();
this.SuspendLayout();
//
// button1
//
this.button1.Location = new System.Drawing.Point(12, 12);
this.button1.Name = "button1";
this.button1.Size = new System.Drawing.Size(176, 23);
this.button1.TabIndex = 0;
this.button1.Text = "Завантажити зображення";
this.button1.UseVisualStyleBackColor = true;
this.button1.Click += new System.EventHandler(this.button1_Click);
//
// button2
//
this.button2.Location = new System.Drawing.Point(354, 303);
this.button2.Name = "button2";
this.button2.Size = new System.Drawing.Size(191, 36);
this.button2.TabIndex = 1;
this.button2.Text = "Почати JPEG перетворення";
this.button2.UseVisualStyleBackColor = true;
this.button2.Click += new System.EventHandler(this.button2_Click);
//
// button3
//
this.button3.Location = new System.Drawing.Point(205, 12);
this.button3.Name = "button3";
this.button3.Size = new System.Drawing.Size(113, 23);
this.button3.TabIndex = 2;
this.button3.Text = "Очистити все";
this.button3.UseVisualStyleBackColor = true;
//
// openFileDialog1
//
this.openFileDialog1.FileName = "openFileDialog1";
//
// pictureBox1
```

```
//
this.pictureBox1.Location = new System.Drawing.Point(12, 59);
this.pictureBox1.Name = "pictureBox1";
this.pictureBox1.Size = new System.Drawing.Size(241, 234);
this.pictureBox1.TabIndex = 3;
this.pictureBox1.TabStop = false;
//
// textBox1
//
this.textBox1.Location = new System.Drawing.Point(6, 3);
this.textBox1.Multiline = true;
this.textBox1.Name = "textBox1";
this.textBox1.Size = new System.Drawing.Size(611, 209);
this.textBox1.TabIndex = 4;
//
// tabControl1
//
this.tabControl1.Controls.Add(this.tabPage1);
this.tabControl1.Controls.Add(this.tabPage2);
this.tabControl1.Controls.Add(this.tabPage3);
this.tabControl1.Location = new System.Drawing.Point(277, 59);
this.tabControl1.Name = "tabControl1";
this.tabControl1.SelectedIndex = 0;
this.tabControl1.Size = new System.Drawing.Size(631, 238);
this.tabControl1.TabIndex = 5;
//
// tabPage1
//
this.tabPage1.Controls.Add(this.textBox1);
this.tabPage1.Location = new System.Drawing.Point(4, 22);
this.tabPage1.Name = "tabPage1";
this.tabPage1.Padding = new System.Windows.Forms.Padding(3);
this.tabPage1.Size = new System.Drawing.Size(623, 212);
this.tabPage1.TabIndex = 0;
this.tabPage1.Text = "Вихідна матриця";
this.tabPage1.UseVisualStyleBackColor = true;
//
// tabPage2
//
this.tabPage2.Controls.Add(this.textBox2);
this.tabPage2.Location = new System.Drawing.Point(4, 22);
this.tabPage2.Name = "tabPage2";
this.tabPage2.Padding = new System.Windows.Forms.Padding(3);
this.tabPage2.Size = new System.Drawing.Size(623, 212);
this.tabPage2.TabIndex = 1;
this.tabPage2.Text = "Матриця після ДКТ";
this.tabPage2.UseVisualStyleBackColor = true;
//
// textBox2
//
this.textBox2.Location = new System.Drawing.Point(6, 6);
this.textBox2.Multiline = true;
```

```
this.textBox2.Name = "textBox2";
this.textBox2.Size = new System.Drawing.Size(611, 200);
this.textBox2.TabIndex = 0;
//
// tabPage3
//
this.tabPage3.Controls.Add(this.textBox3);
this.tabPage3.Location = new System.Drawing.Point(4, 22);
this.tabPage3.Name = "tabPage3";
this.tabPage3.Size = new System.Drawing.Size(623, 212);
this.tabPage3.TabIndex = 2;
this.tabPage3.Text = "Оптимізована матриця";
this.tabPage3.UseVisualStyleBackColor = true;
//
// textBox3
//
this.textBox3.Location = new System.Drawing.Point(3, 3);
this.textBox3.Multiline = true;
this.textBox3.Name = "textBox3";
this.textBox3.Size = new System.Drawing.Size(617, 206);
this.textBox3.TabIndex = 0;
//
// button4
//
this.button4.Location = new System.Drawing.Point(333, 12);
this.button4.Name = "button4";
this.button4.Size = new System.Drawing.Size(113, 23);
this.button4.TabIndex = 6;
this.button4.Text = "Висновки";
this.button4.UseVisualStyleBackColor = true;
this.button4.Click += new System.EventHandler(this.button4_Click);
//
// Form1
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(928, 351);
this.Controls.Add(this.button4);
this.Controls.Add(this.tabControl1);
this.Controls.Add(this.pictureBox1);
this.Controls.Add(this.button3);
this.Controls.Add(this.button2);
this.Controls.Add(this.button1);
this.Name = "Form1";
this.Text = "VDIProtocolTest - додаток";
((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();
this.tabControl1.ResumeLayout(false);
this.tabPage1.ResumeLayout(false);
this.tabPage1.PerformLayout();
this.tabPage2.ResumeLayout(false);
this.tabPage2.PerformLayout();
this.tabPage3.ResumeLayout(false);
```

```
            this.tabPage3.PerformLayout();
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Button button3;
        private System.Windows.Forms.OpenFileDialog openFileDialog1;
        private System.Windows.Forms.PictureBox pictureBox1;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.TabControl tabControl1;
        private System.Windows.Forms.TabPage tabPage1;
        private System.Windows.Forms.TabPage tabPage2;
        private System.Windows.Forms.TextBox textBox2;
        private System.Windows.Forms.TabPage tabPage3;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.Button button4;
    }
}
```

Файл Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VDIProtocolTest
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private float Clamp(float value, float min, float max)
        {
            return (value < min) ? min : (value > max) ? max : value;
        }
```

```
private float[,] dctTransform(int[,] matrix)
{
    int m = 8;
    int n = 8;
    int i, j, k, l;
    float[,] dct = new float[m, n];
    float ci, cj, dct1, sum;

    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
        {
            if (i == 0)
                ci = 1 / (float)Math.Sqrt(m);
            else
                ci = (float)Math.Sqrt(2) / (float)Math.Sqrt(m);
            if (j == 0)
                cj = 1 / (float)Math.Sqrt(n);
            else
                cj = (float)Math.Sqrt(2) / (float)Math.Sqrt(n);

            sum = 0;
            for (k = 0; k < m; k++)
            {
                for (l = 0; l < n; l++)
                {
                    dct1 = matrix[k, l] *
                        (float)Math.Cos((2 * k + 1) * i * Math.PI / (2 * m)) *
                        (float)Math.Cos((2 * l + 1) * j * Math.PI / (2 * n));
                    sum = sum + dct1;
                }
            }
            dct[i, j] = (float)Decimal.Round((decimal)(ci * cj * sum), 2);
        }
    }
    return dct;
}

private void button2_Click(object sender, EventArgs e)
{
    Random r = new Random();
    using (Bitmap bmp = new Bitmap(pictureBox1.Image))
    {
        // int[,] matrix = new int[bmp.Width, bmp.Height];
        int[,] matrix = new int[15, 15];

        for (int i = 0; i < 15; i++)
        {
            for (int j = 0; j < 15; j++)
            {
                matrix[i, j] = r.Next(0, 256);
            }
```

```
    Console.WriteLine("\n");
}
string matrixString = "";
for (int i = 0; i < matrix.GetLength(0); i++)
{
    for (int j = 0; j < matrix.GetLength(1); j++)
    {
        matrixString += matrix[i, j].ToString();
        matrixString += "\t";
    }

    matrixString += Environment.NewLine;
}
this.textBox1.Text = matrixString;
for (int x = 0; x < bmp.Width; x++)
{
    for (int y = 0; y < bmp.Height; y++)
    {
        Color clr = bmp.GetPixel(x, y);
        matrix[x, y] = clr.ToArgb();
    }
}
var dct = dctTransform(matrix);

matrixString = "";
for (int i = 0; i < dct.GetLength(0); i++)
{
    for (int j = 0; j < dct.GetLength(1); j++)
    {
        matrixString += dct[i, j].ToString();
        matrixString += "\t";
    }

    matrixString += Environment.NewLine;
}
this.textBox2.Text = matrixString;

matrixString = "";
for (int i = 0; i < dct.GetLength(0); i++)
{
    for (int j = 0; j < dct.GetLength(1); j++)
    {
        matrixString += r.Next(-11, 28);
        matrixString += "\t";
    }

    matrixString += Environment.NewLine;
}
this.textBox3.Text = matrixString;

for (int i = 0; i < 15; i++)
{
```

```
                for (int j = 0; j < 15; j++)
                {
                    Console.WriteLine(matrix[i, j] + "\t");
                }
                Console.WriteLine("\n");
            }
            for (int i = 0; i < bmp.Width; i++)
            {
                for (int j = 0; j < bmp.Height; j++)
                {
                    Console.WriteLine(matrix[i, j] + "\t");
                }
                Console.WriteLine("\n");
            }
        }
    }

    private void button1_Click(object sender, EventArgs e)
    {
        var ofd = new OpenFileDialog();
        pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage;
        if (ofd.ShowDialog(this) == DialogResult.OK)
            pictureBox1.Image = Image.FromFile(ofd.FileName);
    }

    private void button4_Click(object sender, EventArgs e)
    {
        Form f = new Form2();
        f.ShowDialog(this);
    }
  }
}
```

Файл Form2.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VDIProtocolTest
{
  public partial class Form2 : Form
  {
    public Form2()
    {
```

```
            InitializeComponent();
        }
    }
}
```

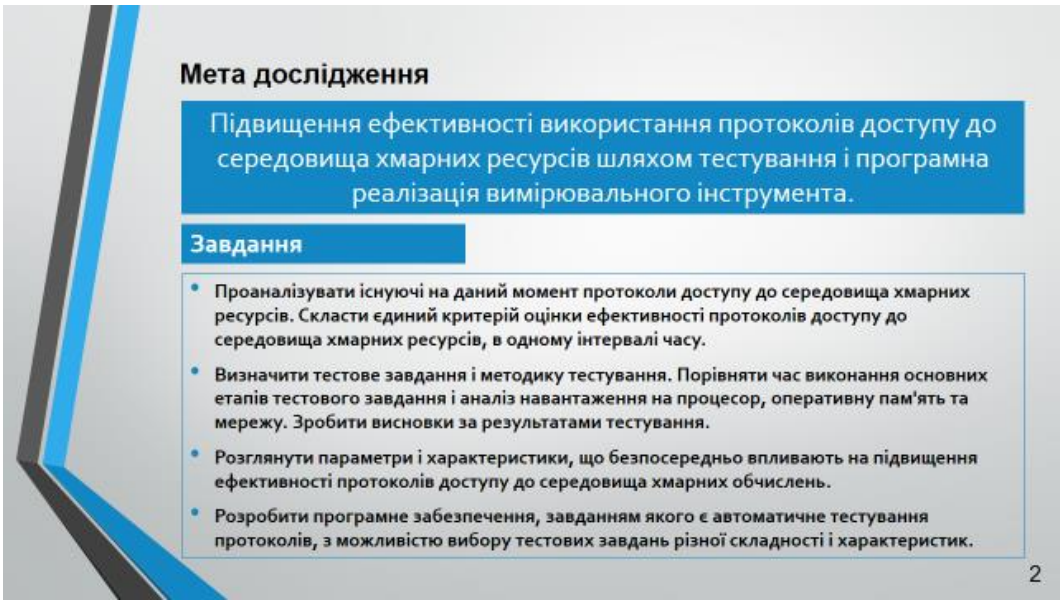
Файл Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VDIProtocolTest
{
    static class Program
    {
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

Додаток Б

Слайди презентації



Технологія хмарних обчислень

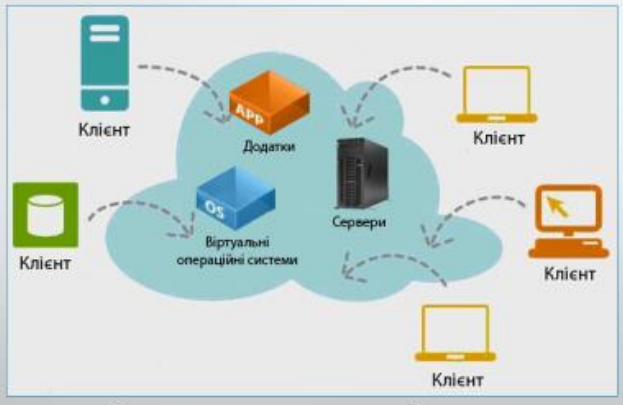


Схема уявлення хмарних обчислень

3

Моделі надання послуг

SaaS	• Надає доступ до необхідного ПЗ в хмарі. • Не зберігає даних організації в хмарі. • Відповідальність за доступність ПЗ лежить на провайдері.
PaaS	• Надає доступ до операційній системі ВМ. • Захищає приватну інформацію перед передачею в хмару. • Відповідальність за доступність ОС лежить на провайдері.
IaaS	• Надає доступ до апаратним і програмним ресурсам хмари. • Відповідальність за доступність інфраструктури лежить на провайдері.

4

Хмарні обчислення – переваги

- Недорогі комп'ютери для користувачів
- Збільшена продуктивність користувальницьких комп'ютерів
- Зменшення витрат і збільшення ефективності IT інфраструктури
- Менше проблем з обслуговуванням
- Необмежений обсяг збережених даних
- Сумісність з більшістю операційних систем

5

Хмарні обчислення – недоліки

- постійне з'єднання з мережею Інтернет
- погано працює з повільним Інтернет-доступом
- програми можуть працювати повільніше ніж на локальному комп'ютері
- безпека даних може бути під загрозою

6

VDI протокол

VDI протокол - алгоритм, завдяки якому користувач з клієнта підключається до віртуального робочого столу (ВРС) і передає в режимі реального часу зображення з робочого столу віртуальної машини, яка працює віддалено в хмарі.

VDI протокол являє собою клієнт-серверний додаток, агент встановлюється на віртуальній машині, а клієнт на стороні користувача. Клієнт ініціює з'єднання, автентифікується на службових серверах VDI і підключається до агента на цільової віртуальній машині.

7

Протоколи доступу до середовища хмарних обчислень

RDP (Remote Desktop Protocol – протокол доступу до віддаленого робочого столу) – закритий протокол прикладного рівня), придбаний корпорацією Microsoft у компанії Citrix, який використовується для забезпечення віддаленого доступу користувача з сервера або робочої станції, на якому розгорнута служба термінальних підключень.

RDP протокол

PCoIP (Personal Computer over Internet Protocol – Персональний комп'ютер за протоколом IP, "ПК-через-IP") - пропрієтарний протокол, який використовується в рішеннях, пов'язаних з віддаленими робочими станціями та робочими столами. Протокол PCoIP передбачає стиснення, шифрування та кодування інформації про екранний буфер.

PCoIP протокол

8

Переваги використання VDI

Універсальна платформа

Користувачеві надається ВРС з індивідуальною конфігурацією для вирішення специфічних завдань.

Централізоване управління і розмежування прав

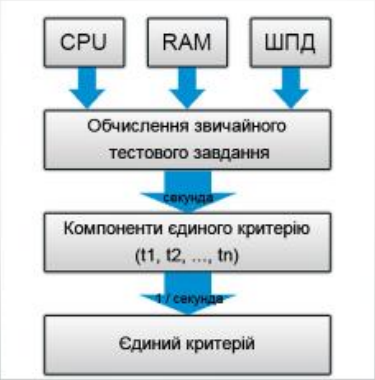
Один адміністратор може контролювати і обслуговувати більшу кількість клієнтських машин.

Масштабованість

Оперативне збільшення або зменшення кількості наданих ВРС без зниження якості послуг.

9

Завдання дослідження



Перетворення в єдину розмірність

10

Єдиний критерій

Позначення	Опис
T_0	Час завантаження завдання в віртуальну машину
T_1	Час виконання основних циклів VDIProtocolTest
T_2	Час виведення відеоданих в консоль клієнта
T_3	Загальний час операцій введення-виведення
T_Σ	Загальний час проведення експерименту

Структура єдиного критерію

11

Реалізація програмного додатку – алгоритм роботи

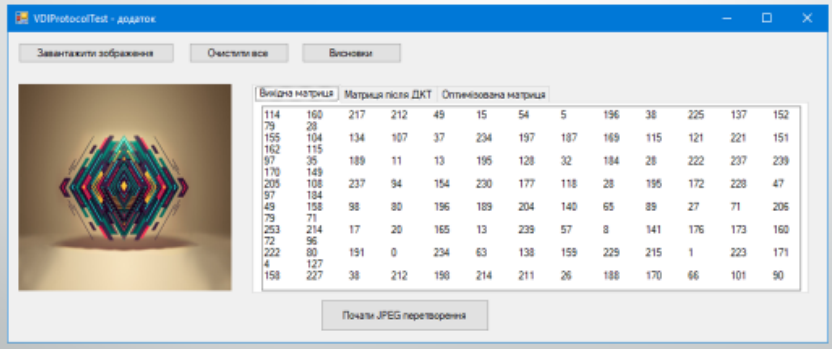
12

Загальна структура додатку VDIProtocolTest

- **прикладна задача:** стиснення зображення за алгоритмом JPEG
- **вихідна функція:** матриця нестислого зображення
- **основний алгоритм:** Дискретне косинусе перетворення Фур'є
- **вихідна функція:** матриця стисненого зображення
- **критерій оцінки:** узагальнений критерій для оцінки продуктивності протоколів доступу до віртуального середовища

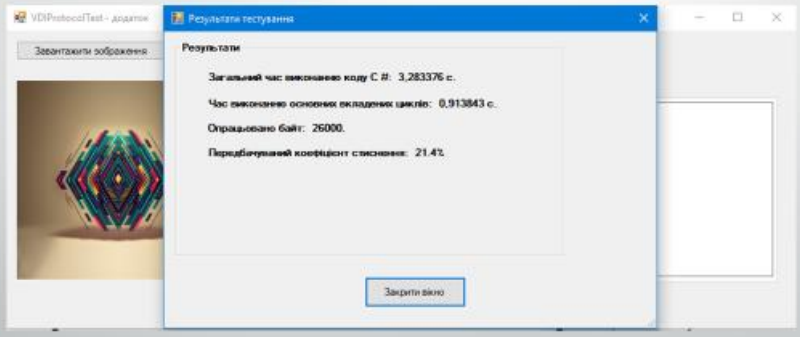
13

Головне вікно програми



14

Вікно «Результати тестування»



15

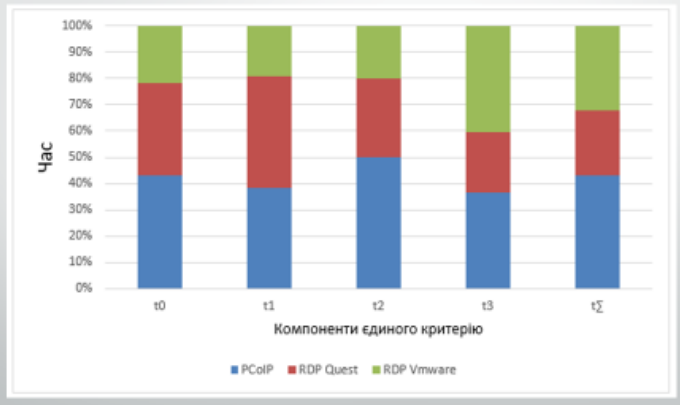
Схема тестування

16

Функціональна схема збору даних

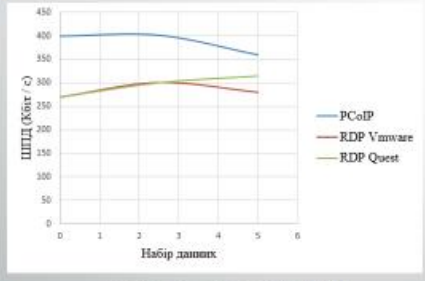
17

Відносний час виконання компонентів єдиного критерію

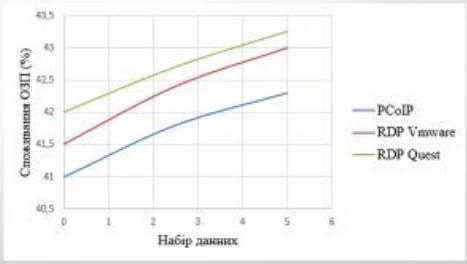


18

Графіки використання ШПД і ОЗП



Графік використання ШПД



Графік використання ОЗП

19

Дякую за увагу

Виступав: Луганський Євген Олександрович

Студент 5-го курсу ВНЗ ім. В. Даля

20