

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної

інженерії та кібербезпеки

Андрія Сфіменко

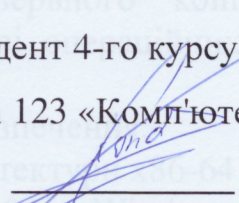
2 лютого 2026 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

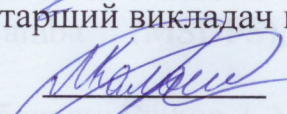
до випускної кваліфікаційної роботи бакалавра
на тему: «Проект розподіленого файлового сховища
організації на базі рішення Samba»

Виконав: студент 4-го курсу групи КІ-22-1

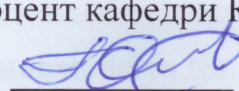
спеціальності 123 «Комп'ютерна інженерія»

 Дмитрій КОНДРАТЮК

Керівник: старший викладач кафедри КІ та КБ,

 Марія КОЛОЩУК

Рецензент: доцент кафедри КІ та КБ, к.пед.н., доцент

 Олена ГОЛОВНЯ

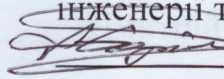
Житомир – 2026

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 123 «Комп'ютерна інженерія»
Освітня програма «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко

2 лютого 2026 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу бакалавра

Студент: Кондратюк Дмитрій Олегович

Тема роботи: «Проект розподіленого файлового сховища організації на базі рішення Samba»

затверджена Наказом університету від 2 лютого 2026 р. №57/с

Термін здачі студентом закінченої роботи 23 червня 2026 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):

Об'єкт дослідження — процес організації, структурування та забезпечення безпечного спільного доступу до корпоративних файлових ресурсів у гетерогенному середовищі підприємства.

Предмет дослідження — методи, технології та програмно-апаратні засоби побудови розподіленого простору імен файлових ресурсів (DFS/MSDFS), розмежування прав доступу, резервного копіювання та безперервного телеметричного моніторингу на базі операційних систем Linux і протоколу SMBv3.

Обладнання та програмне забезпечення:

1. Віртуальні серверні вузли архітектури x86-64 та клієнтські ПК.
2. Операційні системи Ubuntu 25.04 та Windows 10/11
3. Програмний комплекс Samba (MSDFS), утиліти POSIX ACL (setfacl/getfacl).
4. Інструменти моніторингу та бекапу: Python 3, MQTT Mosquitto, Node-RED, Cron.
5. Утиліти тестування продуктивності: dd та fio.

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Фальковський І.Г.	20.02.2026	20.02.2026
2	Фальковський І.Г.	20.02.2026	20.02.2026
3	Фальковський І.Г.	20.02.2026	20.02.2026

Студент Дмитрій КОНДРАТЮК

Керівник Марія КОЛОЩУК

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання, аналіз вимог та огляд літератури	02.02 – 12.02	Виконано
2	Проектування логічної та фізичної архітектури розподіленої системи	13.02 – 26.02	Виконано
3	Розгортання та базова конфігурація серверів Samba і DFS (MSDFS)	27.02 – 12.03	Виконано
4	Впровадження моделі безпеки на основі масок Samba та POSIX ACL	13.03 – 26.03	Виконано
5	Розробка агента моніторингу метрик апаратних ресурсів (Python, psutil)	27.03 – 09.04	Виконано
6	Інтеграція підсистеми передачі телеметрії через MQTT (Mosquitto, Node-RED)	10.04 – 23.04	Виконано
7	Налаштування автоматичного резервного копіювання за розкладом (Cron)	24.04 – 07.05	Виконано
8	Проведення навантажувального тестування продуктивності (dd, fio)	08.05 – 21.05	Виконано
9	Оформлення пояснювальної записки та аналіз результатів тестування	22.05 – 04.06	Виконано
10	Підготовка презентації, ілюстративного матеріалу та доповіді	05.06 – 18.06	Виконано
11	Захист	26.06	

Студент Дмитрій КОНДРАТЮК

Керівник Марія КОЛОЩУК

АНОТАЦІЯ

Кваліфікаційна робота присвячена проектуванню, розгортанню, захисту та комплексному тестуванню логічно централізованої розподіленої файлової інфраструктури з підсистемою резервного копіювання для підприємства. Пояснювальна записка складається зі 97 сторінок, містить 49 рисунків, 20 таблиць та 22 найменування у списку використаних джерел.

Метою випускної кваліфікаційної роботи є теоретичне обґрунтування, практична реалізація та експериментальний аналіз ефективності безпечного розподіленого середовища збереження та обміну даними з автоматизованим резервним копіюванням і моніторингом на базі операційних систем Linux із використанням технології простору імен DFS та протоколу SMBv3.

У межах роботи проаналізовано сучасні підходи до побудови корпоративних сховищ даних та обґрунтовано вибір системного стеку технологій на основі Ubuntu Server та Samba. Визначено основні архітектурні вимоги для забезпечення прозорості централізації мережеских каталогів підрозділів. Показано розробку і конфігурування логічного простору імен, який дозволяє повністю приховати від кінцевих користувачів фізичну розрізненість серверних вузлів. Розглянуто особливості побудови двоконтурного безпекового периметра на основі авторизаційних директив Samba та рекурсивних списків контролю доступу POSIX ACL для ізоляції конфіденційної інформації. Розроблено та інтегровано підсистему автоматичного щоденного резервного копіювання за розкладом за допомогою Cron та фоновий клієнтський агент мовою Python 3 для збору метрик утилізації апаратних ресурсів з передачею телеметрії через MQTT-брокер Mosquitto на інтерактивний веб-дашборд Node-RED.

					KKIKB.KP.B – 123 – 07 – ПЗ		
Зм.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Кондратюк Д.О.		24.06.26	Проект розподіленого файлового сховища організації на базі рішення Samba	Літ.	Арк.
Керівник		Колощук М.С.		24.06.26			4
Реценз.		Головня О.С.		24.06.26			98
Н. контр.		Колощук М.С.		24.06.26		ФІКТ, Гр. КІ-22-1	
Затверд.		Єфіменко А.А.		24.06.26			

Наведено результати функціонального та навантажувального тестування продуктивності дискової й мережевої підсистем за допомогою утиліт dd та fio, що дозволило зафіксувати реальні показники пропускної здатності, кількості операцій введення-виведення та часових затримок системи під стресовим навантаженням. Сформувано комплексні інженерні рекомендації щодо подальшого масштабування сховища шляхом впровадження SSD-кешування та міграції на Samba4 Active Directory Domain Controller.

Ключові слова: РОЗПОДІЛЕНА ФАЙЛОВА СИСТЕМА, SAMBA, MSDFS, SMBV3, UBUNTU SERVER, POSIX ACL, РЕЗЕРВНЕ КОПЮВАННЯ, МОНІТОРИНГ, MQTT, NODE-RED, ТЕСТУВАННЯ ПРОДУКТИВНОСТІ, IOPS, LATENCY.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

ANNOTATION

The bachelor's thesis is dedicated to the design, deployment, security, and comprehensive testing of a logically centralized distributed file infrastructure with a data backup subsystem for an enterprise. The explanatory note consists of 97 pages, contains 49 figures, 20 tables, and 22 titles in the list of used sources.

The **aim** of the qualification work is the theoretical substantiation, practical implementation, and experimental efficiency analysis of a secure distributed data storage and exchange environment based on Linux operating systems using DFS namespace technology and the SMBv3 protocol.

Within the scope of the work, modern approaches to building corporate data storages are analyzed and the choice of the system technology stack based on Ubuntu Server and Samba is substantiated. The main architectural requirements to ensure transparent centralization of network directories of departments are determined. The development and configuration of a logical namespace, which allows completely hiding the physical fragmentation of server nodes from end users, is shown. The features of building a two-circuit security perimeter based on Samba authorization directives and recursive POSIX ACL access control lists for isolating confidential information are considered. A subsystem of automatic daily scheduled backup using Cron and a background client agent in Python 3 for collecting hardware resource utilization metrics with telemetry transmission via the Mosquitto MQTT broker to the Node-RED interactive web dashboard was developed and integrated. The results of functional and stress testing of disk and network subsystem performance using dd and fio utilities are presented, which allowed capturing real indicators of bandwidth, the number of input-output operations, and system time latencies under stress load. Comprehensive engineering recommendations for further scaling of the storage by introducing SSD caching and migrating to Samba4 Active Directory Domain Controller are formed.

Keywords: DISTRIBUTED FILE SYSTEM, SAMBA, MSDFS, SMBV3, UBUNTU SERVER, POSIX ACL, BACKUP, MONITORING, MQTT, NODE-RED, PERFORMANCE TESTING, IOPS, LATENCY.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ВСТУП	10
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ	13
1.1 Проблематика організації файлового зберігання на підприємстві	13
1.2 Огляд протоколів спільного доступу до файлів SMB/CIFS, NFS, FTP/SFTP, WebDAV — порівняння, сфери застосування	17
1.3 Аналіз технологій та рішень для файлових сховищ	22
1.4 Огляд аналогічних рішень та існуючих реалізацій	28
1.5 Постановка задачі та вимоги до системи	31
РОЗДІЛ 2. МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ ФАЙЛОВОГО СХОВИЩА	35
2.1 Розробка архітектури системи	35
2.2 Моделювання мережевої інфраструктури організації	38
2.3 Проєктування структури файлового сховища та DFS	41
2.4 Проєктування моделі розмежування прав доступу та автентифікації	44
2.5. Моделювання підсистеми резервного копіювання	47
РОЗДІЛ 3. РОЗГОРТАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ	51
3.1 Встановлення та базове налаштування Samba	51
3.2 Налаштування DFS та єдиного простору імен	55
3.3 Налаштування автентифікації та прав доступу	59
3.4 Налаштування та тестування підсистеми моніторингу	61
3.5 Налаштування підсистеми резервного копіювання даних	65
3.6 Функціональне та навантажувальне тестування інфраструктури	68
3.7 Аналіз результатів та оцінка ефективності реалізації	74
ВИСНОВКИ	78
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	80
ДОДАТКИ	82

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ACL (Access Control List) – список контролю доступу

API (Application Programming Interface) – інтерфейс прикладного програмування

ASCII (American Standard Code for Information Interchange) – американський стандартний код для обміну інформацією

BW (Bandwidth) – пропускна здатність мережі або дискової підсистеми

CPU (Central Processing Unit) – центральний процесор

DFS (Distributed File System) – розподілена файлова система

GB (Gigabyte) – гігабайт

Gbps (Gigabits per second) – гігабіт за секунду

IOPS (Input/Output Operations Per Second) – кількість операцій введення-виведення за секунду

IP (Internet Protocol) – міжмережевий протокол

JSON (JavaScript Object Notation) – текстовий формат обміну даними, заснований на JavaScript

LTS (Long Term Support) – довгострокова підтримка операційної системи

MB/s (Megabytes per second) – мегабайт за секунду

MSDFS (Microsoft Distributed File System) – реалізація розподіленої файлової системи від Microsoft, підтримувана в Samba

MQTT (Message Queuing Telemetry Transport) – легковажний мережевий протокол для передачі телеметрії

NAS (Network Attached Storage) – мережеве сховище даних

ОЗУ – оперативний запам'ятовуючий пристрій (оперативна пам'ять)

ОС – операційна система

ПК – персональний комп'ютер

ПЗ – програмне забезпечення

POSIX (Portable Operating System Interface) – набір стандартів сумісності переносних операційних систем

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

QoS (Quality of Service) – якість мережевого обслуговування та пріоритезації

RAM (Random Access Memory) – оперативна пам'ять комп'ютера

SMB (Server Message Block) – мережевий протокол прикладного рівня для спільного доступу до файлів

SSD (Solid-State Drive) – твердотільний накопичувач

SSH (Secure Shell) – мережевий протокол захищеного віддаленого керування ОС

TCP (Transmission Control Protocol) – протокол керування передачею даних

UDP (User Datagram Protocol) – протокол користувацьких датаграм

UI (User Interface) – інтерфейс користувача

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колощук М.С.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. У сучасних умовах розвитку корпоративних інформаційних систем ефективне керування великими масивами даних є одним із головних чинників стабільної роботи будь-якого підприємства. Проте на практиці більшість організацій стикається з децентралізацією файлових ресурсів. Ситуація, коли документи, звіти та робочі матеріали зберігаються розрізнено на локальних комп'ютерах співробітників або окремих некерованих накопичувачах, створює суттєві труднощі. Це призводить до дублювання інформації, нераціонального використання дискового простору, а головне — значно ускладнює централізоване керування правами доступу та регулярне резервне копіювання, що створює серйозні ризики для інформаційної безпеки.

Традиційне вирішення цієї проблеми на базі пропрієтарного програмного забезпечення, зокрема ліцензійних платформ Windows Server, наразі супроводжується економічними та стратегічними ризиками. Сучасні тенденції розвитку ІТ-сфери вимагають від вітчизняних підприємств поступового відходу від закритих комерційних рішень на користь програмного забезпечення з відкритим вихідним кодом. Використання комерційних закордонних продуктів сьогодні пов'язане не лише з високою вартістю ліцензування, але й з імовірністю раптового припинення технічної підтримки чи обмеження доступу до сервісів, що змушує шукати надійні та незалежні альтернативи.

Найбільш раціональним розв'язанням цієї задачі є впровадження програмного комплексу Samba у поєднанні з технологією розподіленої файлової системи DFS. Таке поєднання дозволяє розгорнути повноцінне сховище на базі операційної системи Linux, забезпечуючи сумісність із клієнтськими робочими станціями як під керуванням Windows, так і Linux. Використання технології DFS дає змогу об'єднати фізично розрізнені сервери підприємства в єдиний віртуальний простір імен. Для кінцевого користувача вся структура сховища виглядає як одна спільна мережева папка, незалежно від фактичного розташування файлів на дисках. Це дозволяє оптимізувати витрати на ІТ-інфраструктуру, підвищити безпеку даних

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

та забезпечити зручність адміністрування, що й зумовлює актуальність обраної теми дослідження.

Метою кваліфікаційної роботи є проєктування та розгортання централізованого файлового сховища організації з єдиним простором імен на базі Samba і DFS для ліквідації проблеми децентралізації корпоративних даних, забезпечення гнучкого розмежування прав доступу в гетерогенному середовищі, оптимізації витрат на IT-інфраструктуру, а також створення супутнього комплексу автоматизованого резервного копіювання даних та моніторингу апаратних ресурсів серверів.

Визначена мета обумовлює наступні завдання:

- дослідити проблематику організації файлового зберігання на підприємствах, провести порівняльний аналіз протоколів спільного доступу та обґрунтувати вибір програмного комплексу Samba та технології DFS;
- розробити архітектурну модель централізованого сховища та спроектувати структуру віртуального простору імен DFS і модель розмежування прав доступу користувачів;
- практично розгорнути та налаштувати сервери Samba, конфігурувати єдиний простір імен DFS та реалізувати систему автентифікації користувачів і груп;
- розробити та впровадити стратегію автоматизованого резервного копіювання даних та інтегрувати систему моніторингу метрик серверної інфраструктури в реальному часі;
- провести функціональне тестування створеного сховища, оцінити його продуктивність за допомогою спеціалізованих утиліт і проаналізувати відповідність отриманих результатів поставленим вимогам.

Об'єктом дослідження є процес організації, структурування та забезпечення безпечного спільного доступу до корпоративних файлових ресурсів у гетерогенному середовищі підприємства. Цей процес розглядається як комплексний елемент IT-інфраструктури, що охоплює організаційно-технічні заходи з централізації інформаційних масивів, забезпечення ефективної взаємодії між різнотипними (гетерогенними) клієнтськими робочими станціями,

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

автоматизацію процесів збереження цілісності даних, а також безперервний адміністративний контроль за станом, доступністю та безпекою мережесховищ з метою підтримки стабільності бізнес-процесів компанії.

Предметом дослідження є архітектурні методи, програмні засоби та практичні технології побудови централізованих файлових сховищ на базі програмного комплексу Samba, технології простору імен DFS та протоколу SMBv3.

У межах об'єкта дослідження основна увага зосереджена на механізмах логічного об'єднання фізично розрізнених серверів і дискових ресурсів у єдиний віртуальний простір імен (компонент DFS namespaces), алгоритмах автентифікації користувачів, гнучкому розмежуванні прав доступу на рівні мережесховищ та файлової системи, а також на інструментах і методах проектування супутньої підсистеми автоматизованого моніторингу, що забезпечує збір телеметрії (навантаження на процесор, утилізація оперативної пам'яті, заповнення дискового простору) та візуалізацію стану апаратної частини серверів у реальному часі.

У процесі виконання кваліфікаційної роботи було розроблено проект безпечної, відмовостійкої та логічно централізованої розподіленої файлової інфраструктури підприємства на базі операційної системи Ubuntu Server та сервісів Samba DFS (MSDFS). Комплекс забезпечує прозоре об'єднання фізично розрізнених серверів сховища в єдину точку входу з дворівневим безпековим контуром розмежування доступу на основі POSIX ACL, а також включає інтегровані автономні підсистеми щоденного резервного копіювання через Cron та безперервного телеметричного моніторингу апаратних ресурсів за протоколом MQTT з візуалізацією в Node-RED.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Проблематика організації файлового зберігання на підприємстві

Ефективне функціонування сучасного підприємства безпосередньо залежить від швидкості, надійності та безпеки обміну інформацією між його структурними підрозділами. Основним компонентом цього обміну є корпоративні файлові масиви, що містять фінансову документацію, технічні проекти, юридичні договори та інші критично важливі дані. Проте в процесі природного розвитку та масштабування ІТ-інфраструктури більшість організацій стикаються із серйозними архітектурними проблемами в організації збереження файлів, які за відсутності централізованого підходу переростають у системну кризу супроводу мережі.

Першою і найбільш очевидною проблемою є децентралізація та розрізненість мережесих ресурсів. На початкових етапах розвитку компанії спільний доступ до файлів часто організовується хаотично: створюються окремі локальні папки на робочих станціях провідних спеціалістів або купуються автономні мережеві накопичувачі (NAS) початкового рівня для конкретних відділів. У результаті інформаційний простір підприємства перетворюється на ізольовані «острівці даних». Співробітники змушені витратити значний робочий час на пошук потрібного документа, оскільки актуальна інформація може зберігатися на будь-якому з десятків мережесих вузлів, адреси яких не мають єдиної логіки.

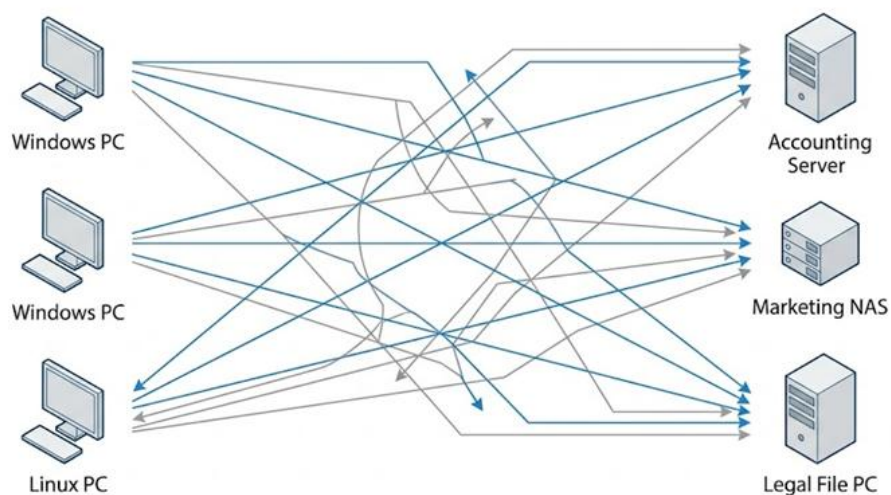


Рисунок 1.1 – Архітектура децентралізованого зберігання даних на підприємстві

З розрізненістю ресурсів нерозривно пов'язане неконтрольоване дублювання даних (надмірність інформації). Коли працівники різних підрозділів не мають

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

спільного координаційного центру для роботи над проектами, вони створюють копії одних і тих самих файлів у своїх локальних сховищах. Це призводить до появи десятків версій одного документа. Окрім критичного ускладнення колективної роботи та виникнення помилок через використання застарілих даних, це явище спричиняє нераціональне використання дискового простору серверів, штучно збільшуючи витрати на закупівлю нових носіїв інформації.

Це призводить до появи десятків версій одного документа, які з часом починають функціонувати як окремі файли. Окрім критичного ускладнення колективної роботи, коли виникають серйозні помилки через використання співробітниками застарілих або неактуальних даних, це явище спричиняє швидке і нераціональне використання дискового простору серверів. Організація змушена штучно та неефективно збільшувати капітальні витрати на регулярну закупівлю нових носіїв інформації та розширення масивів зберігання, замість оптимізації існуючого простору.

Таблиця 1.1

Вплив надмірності та децентралізації даних на операційні показники підприємства

Параметр інфраструктури	Стан при децентралізованому підході	Наслідки для бізнес-процесів	Оптимальний цільовий показник
Ефективність дискового простору	Низька (до 40% займають дублікати файлів)	Передчасні витрати на закупівлю жорстких дисків	Коефіцієнт дублювання близький до 0%
Час пошуку інформації	Високий (від 10 до 30 хвилин на один документ)	Зниження загальної продуктивності праці персоналу	Миттєвий доступ через єдину точку
Версійність документів	Неконтрольована (ризик використання застарілих даних)	Фінансові та юридичні помилки у звітності	Наявність єдиного актуального файлу-оригіналу

Найбільш критичною проблемою з точки зору кібербезпеки та захисту інформації є висока складність керування правами доступу та розмежування повноважень. За умов відсутності централізованої системи автентифікації, єдиного каталогу користувачів та спільної політики безпеки, системний адміністратор змушений налаштовувати локальні облікові записи та права доступу індивідуально на кожному окремому сховищі чи сервері.

Коли штат компанії розширюється, ручне керування перетворюється на неконтрольований процес. При звільненні співробітника, зміні його посадових обов'язків або переведенні в інший департамент виникає висока ймовірність людської помилки: адміністратор може просто забути деактивувати обліковий запис на одному з віддалених NAS-накопичувачів. Це створює прямі передумови для внутрішніх витоків конфіденційної інформації (інсайдерські загрози), а також полегшує зловмисникам реалізацію кібератак типу *Lateral Movement* (горизонтальне переміщення) у разі первинної компромізації хоча б однієї робочої станції користувача.

Ситуація додатково ускладнюється повною відсутністю єдиної точки входу для користувачів корпоративної системи. Щоб отримати доступ до необхідних робочих папок різних департаментів, співробітник змушений монтувати (підключати) в операційній системі кілька різних мережевих дисків, кожен з яких жорстко прив'язаний до окремої IP-адреси або імені конкретного сервера.

Для сучасного гетерогенного середовища, де в межах однієї локальної мережі одночасно використовуються робочі станції під керуванням різних операційних систем (сімейства Microsoft Windows, дистрибутивів Linux Ubuntu/Debian, а також macOS), це створює значний експлуатаційний бар'єр. Користувачі постійно плутаються в мережевих шляхах, синтаксис підключення папок відрізняється, а служба технічної підтримки та системні адміністратори витрачають значну кількість робочого часу на рутинне ручне налаштування та відновлення підключень мережевих дисків на комп'ютерах користувачів.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		15

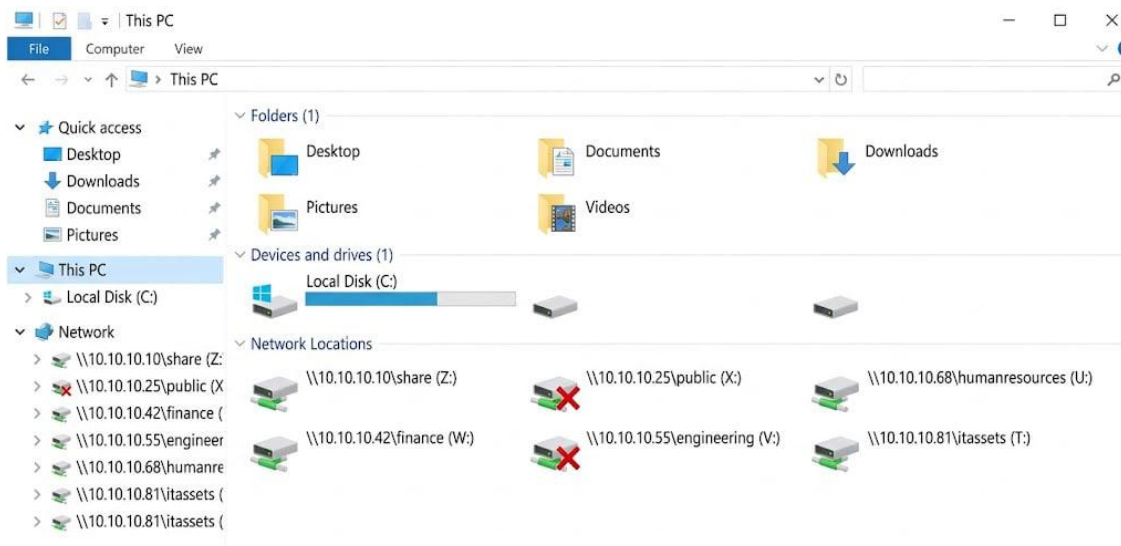


Рисунок 1.2 – Візуалізація перевантаження клієнтського інтерфейсу при підключенні розрізнених мережеских ресурсів

Останнім за переліком, але найважливішим за рівнем потенційної загрози є фактор унеможливлення організації регулярного, цілісного та якісного резервного копіювання (бекапу). Створення надійної стратегії резервування даних у децентралізованій мережі перетворюється на надскладну і неефективну інженерну задачу. Адміністратор змушений розробляти, тестувати та постійно підтримувати десятки розрізнених автоматизованих скриптів для зняття копій з кожного ізольованого сервера окремо.

Через таку децентралізацію часто виникають критичні ситуації, коли важливі папки або нові каталоги якогось із відділів повністю випадають із загального контуру резервування через банальну відсутність координації. У випадку раптового апаратного збою жорстких дисків (наприклад, виходу з ладу RAID-масиву), атаки вірусів-вимагачів (Ransomware) або форс-мажорних обставин, організація ризикує незворотно втратити свою інтелектуальну власність, бази даних, клієнтські списки та фінансову історію, що може призвести до повного краху підприємства на ринку.

Таблиця 1.2

Систематизація ризиків та вразливостей децентралізованої моделі зберігання

Категорія ризику	Конкретний прояв у децентралізованому середовищі	Потенційні збитки для підприємства	Технічний спосіб нівелювання ризику
Інформаційна безпека	Наявність залишкових прав звільнених працівників на розрізнених вузлах	Витік комерційних даних, порушення конфіденційності	Централізована автентифікація, єдина політика доступу
Апаратна відмова	Відсутність бекапу на ізольованих робочих станціях та NAS	Незворотна втрата документів, зупинка бізнес-процесів	Автоматизований бекап з єдиного сховища
Адміністрування	Необхідність ручного налаштування кожного мережевого диска	Великі витрати часу ІТ-відділу, людський фактор	Впровадження віртуального простору імен DFS
Моніторинг	Відсутність контролю за утилізацією дисків та процесорів	Раптова відмова вузлів через переповнення пам'яті або дисків	Інтеграція легковажних агентів збору телеметрії через MQTT

Додатковим ускладнюючим фактором є відсутність превентивного контролю за станом апаратних ресурсів. Коли сервери розподілені по різних кабінетах або сегментах мережі без єдиної системи моніторингу, адміністратор дізнається про проблему (наприклад, критичне вичерпання вільного місця або перегрів процесора) вже за фактом відмови обладнання.

Таким чином, детальний аналіз предметної області чітко доводить, що подальше екстенсивне нарощування дискових потужностей без корінної зміни архітектури системи є тупиковим шляхом. Організації необхідний перехід до сучасної централізованої моделі зберігання даних з логічним об'єднанням усіх ресурсів у єдиний простір, який забезпечить прозоре кросплатформне адміністрування, надійний захист інформації та автоматизований моніторинг стану серверів у реальному часі.

1.2 Огляд протоколів спільного доступу до файлів SMB/CIFS, NFS, FTP/SFTP, WebDAV — порівняння, сфери застосування

Проектування сучасної інфраструктури централізованого зберігання даних вимагає від інженера глибокого системного аналізу мережевих протоколів прикладної моделі OSI [16], які забезпечують віддалений доступ та маніпуляцію файловими об'єктами. Вибір базового протоколу визначає не лише кінцеву

швидкість передачі даних, але й рівень навантаження на процесори серверів, гнучкість інтеграції із системами каталогу користувачів, а також стабільність підключення клієнтських робочих станцій. На сьогоднішній день в ІТ-індустрії сформувався чіткий перелік стеків протоколів, що використовуються для організації мережевої взаємодії з файлами: SMB/CIFS, NFS, FTP/SFTP та WebDAV. Кожен із них розроблявся під конкретні технологічні завдання, тому має унікальні архітектурні особливості, переваги та експлуатаційні обмеження.

Історично першим масштабним стандартом для персональних комп'ютерів став протокол SMB (Server Message Block), розроблений компанією IBM, а згодом глибоко модернізований корпорацією Microsoft. У певний період розвитку (кінець 1990-х років) його версія під назвою CIFS (Common Internet File System) позиціонувалася як універсальний протокол для Інтернету [17], проте через надмірну завантаженість службовими пакетами (так звану «балакучість» протоколу) та низьку безпеку, CIFS визнано застарілим і повністю виведено з експлуатації. Сучасні системи використовують стандарти SMBv2 та, переважно, SMBv3.

Архітектура SMBv3 є високооптимізованою: вона підтримує шифрування даних «на льоту» за допомогою алгоритмів AES-128/256-GCM, механізм SMB Multichannel (одночасне агрегування кількох мережевих каналів для збільшення швидкості), а у високопродуктивних середовищах із відповідним обладнанням — і технологію SMB Direct (RDMA) [9]. SMB орієнтований на роботу всередині локальних мереж (LAN) через TCP-порт 445 і забезпечує унікальну функцію [17] опорного блокування файлів (Oplocks / Lease Mechanisms) [8]. Це дозволяє клієнту кешувати частини файлу локально, мінімізуючи мережевий трафік, але миттєво синхронізувати зміни, якщо інший користувач намагається відкрити той самий документ.

На противагу орієнтованому на Windows протоколу SMB, у UNIX-середовищах стандартом де-факто став протокол NFS (Network File System), створений компанією Sun Microsystems [12]. На сьогодні актуальними є версії NFSv3 та NFSv4. Архітектура NFS кардинально відрізняється від SMB: вона

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		18

розроблялася як максимально проста система без збереження стану (stateless) у версії v3, де сервер не зберігає інформацію про поточний стан клієнта. Це забезпечує мінімальні накладні витрати обчислювальних ресурсів і колосальну швидкість читання та запису дрібних файлів, що робить NFS ідеальним для зв'язку між Linux-серверами, кластерами віртуалізації (наприклад, Proxmox або VMware) та високопродуктивними обчислювальними вузлами.

Проте NFS має критичні недоліки при роботі з користувацькими робочими станціями. По-перше, класична автентифікація в NFSv3 базується виключно на ідентифікаторах користувача (UID) операційної системи Linux. Якщо злоумисник отримає права root на своєму комп'ютері та підробить UID, він отримає повний доступ до чужих файлів на сервері. У версії NFSv4 цю проблему частково вирішили інтеграцією із захищеним протоколом Kerberos, але це значно ускладнило адміністрування. По-друге, механізми блокування файлів в NFS працюють менш стабільно, ніж в SMB, що часто призводить до пошкодження баз даних або офісних документів при спробі одночасного редагування кількох користувачами.

Для перенесення великих об'єктів на значні відстані через глобальні мережі традиційно використовуються протоколи FTP (File Transfer Protocol) та SFTP (SSH File Transfer Protocol). Попри схожість назв, вони мають абсолютно різну інженерну природу. Класичний FTP працює через два окремі порти (21 для команд і 20 для даних), передає паролі та інформацію у відкритому незашифрованому вигляді, а також має величезні проблеми з проходженням через міжмережеві екрани (Firewalls) через динамічне виділення портів у пасивному режимі. SFTP є розширенням протоколу SSH (Secure Shell), працює через один єдиний TCP-порт 22 і забезпечує повне криптографічне шифрування всього трафіку [18].

Головне обмеження FTP/SFTP полягає в тому, що вони розроблялися як протоколи передачі, а не спільної роботи. Користувач не може відкрити текстовий файл розміром 5 ГБ безпосередньо з SFTP-сервера, змінити там один рядок і зберегти — клієнтське програмне забезпечення змушене спочатку повністю скачати файл у тимчасову локальну директорію, внести зміни, а потім завантажити

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		19

весь об'єкт назад на сервер. Це унеможлиблює організацію колективного папочного простору для офісних працівників.

Специфічним рішенням є протокол WebDAV (Web Distributed Authoring and Versioning), який є надбудовою над стандартним веб-протоколом HTTP/HTTPS. Він працює через стандартні веб-порти 80 та 443, що дає йому унікальну перевагу: трафік WebDAV безперешкодно проходить крізь будь-які корпоративні брандмауери та проксі-сервери, а підключитися до сховища можна з будь-якої точки світу навіть через звичайний веб-браузер. WebDAV активно використовується в сучасних комерційних хмарах (наприклад, Nextcloud або Owncloud).

Однак, якщо аналізувати застосування WebDAV всередині локальної мережі підприємства, він демонструє найнижчі показники продуктивності серед усіх конкурентів. Накладні витрати текстового протоколу HTTP (величезні за обсягом заголовки пакетів, кодування бінарних файлів у текстовий формат XML для передачі метаданих) створюють колосальне навантаження на процесори серверів. Швидкість копіювання файлів у WebDAV у кілька разів нижча, ніж у SMB, а затримки (Latency) при відкритті структури великих каталогів є занадто високими для комфортної щоденної роботи персоналу.

Таблиця 1.3

Комплексний порівняльний аналіз мережевих протоколів доступу до файлів

Технічний критерій порівняння	Протокол SMBv3	Протокол NFSv4	Протокол SFTP	Протокол WebDAV
Базовий транспортний порт	TCP 445	TCP 2049	TCP 22 (через SSH)	TCP 80 / 443
Тип блокування файлів	Опорне (Oplocks/Leases)	Зі збереженням стану (stateful locking)	Відсутнє (тільки повне перезаписування)	Блокування через ексклюзивні токени
Механізм шифрування	Вбудований (AES-GCM)	Потребує Kerberos	Вбудований (SSH-криптографія)	Зовнішній (через SSL/TLS сертифікати)
Основна сфера ефективного застосування	Локальні мережі організацій, гетерогенні середовища	Кластери віртуалізації, дата-центри, Linux-сервери	Віддалене адміністрування, безпечна передача великих файлів	Хмарні сховища, доступ через глобальну мережу Інтернет

Обґрунтування вибору протоколу SMB/CIFS для гетерогенного (Windows + Linux) середовища підприємства є ключовою архітектурною точкою даної роботи. Термін «гетерогенне середовище» означає, що корпоративна мережа є змішаною: серверна частина розгортається на базі операційних систем сімейства Linux з міркувань максимальної стабільності, відсутності ліцензійних витрат та високого рівня базового захисту, тоді як на 90–95% робочих станцій кінцевих користувачів встановлено операційні системи Microsoft Windows, які є стандартом для офісного діловодства.

У таких умовах використання протоколу NFS є недоцільним, оскільки клієнтські версії Windows (Home, Pro) не мають вбудованої якісної підтримки NFS «з коробки», а встановлення сторонніх NFS-клієнтів викликає постійні збої з кодуванням кирилических символів у назвах файлів та конфлікти у правах доступу POSIX. Протоколи SFTP та WebDAV, як було доведено вище, не здатні забезпечити прозоре монтування мережеских дисків із високою швидкістю та коректною логікою блокування файлів під час одночасної роботи офісу.

Протокол SMBv3 є єдиним технологічним рішенням, яке повністю задовольняє вимоги обох сторін гетерогенної мережі. З боку клієнтів (Windows) цей протокол є рідним, він інтегрований у ядро ОС, не потребує встановлення жодного додаткового софту і дозволяє користувачам працювати з мережевими папками через звичайний "Провідник" так само легко, як із локальними дисками. З боку серверів (Linux) програмний комплекс Samba бездоганно емує поведінку Windows-сервера, трансліюючи мережеві виклики протоколу SMB у стандартні системні виклики ядра Linux [6].

Крім того, саме SMBv3 дозволяє побудувати надійну модель безпеки шляхом трансляції розширених прав доступу Windows ACL у права доступу файлової системи Linux (POSIX ACL), зберігаючи повну ідентичність правил безпеки для обох платформ. Це дозволяє оптимізувати капітальні витрати компанії, повністю виключивши витрати на дорогі серверні ліцензії Microsoft, при цьому забезпечуючи користувачам звичний, зручний та високошвидкісний функціонал для роботи з документами.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

SMBv3 Protocol Cross-platform Interaction in Heterogeneous Enterprise Network

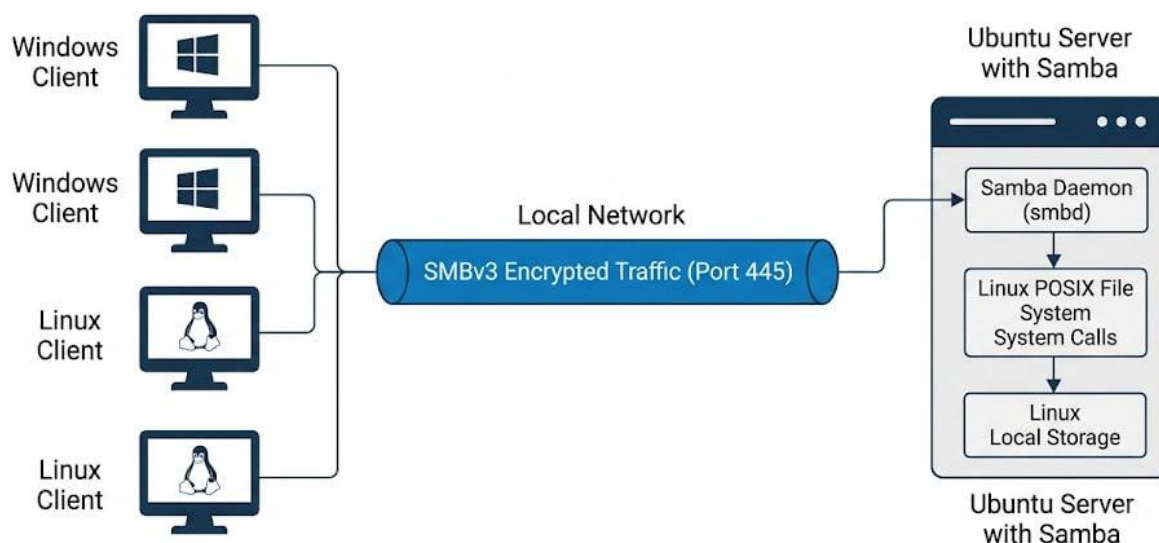


Рисунок 1.3 – Схема трансляції мережеских запитів протоколу SMBv3 у гетерогенному середовищі підприємства

1.3 Аналіз технологій та рішень для файлових сховищ

При виборі технологічної платформи для реалізації централізованого файлового сховища організації зазвичай розглядають два протилежні підходи: пропріетарні закриті системи та рішення з відкритим вихідним кодом. Протягом тривалого часу стандартом у корпоративному сегменті вважалося використання Windows File Server, який є частиною комерційної екосистеми Microsoft. Головна перевага такого підходу традиційно полягала у наявності готових графічних оснасток для швидкого конфігурування прав доступу, що проте жорстко прив'язувало інфраструктуру до екосистеми одного вендора. Зворотною стороною пропріетарного рішення від Microsoft є критично висока вартість володіння, оскільки організація змушена купувати ліцензію на саму серверну операційну систему, а також окремі клієнтські ліцензії доступу (CAL) на кожного співробітника чи пристрій підприємства. Це створює суттєве фінансове навантаження під час масштабування компанії, коли збільшення штату автоматично вимагає додаткових бюджетних асигнувань на програмне забезпечення, що не додає системі нової функціональності, а лише легалізує її

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				22
Змн.	Арк.	№ докум.	Підпис	Дата		

використання. На противагу цьому, використання автономних серверних вузлів на базі дистрибутивів Linux із розгортанням програмного комплексу Samba дозволяє повністю нівелювати ліцензійні витрати, забезпечуючи гнучке локальне керування безпекою без прив'язки до сторонніх платформ.

Альтернативним комерційним варіантом є впровадження готових апаратних рішень у вигляді мережевих накопичувачів NAS від таких вендорів, як Synology або QNAP. Вони постачаються зі спеціалізованою операційною системою на базі модифікованого ядра Linux, мають зручні веб-інтерфейси та вбудовані фірмові утиліти для швидкого розгортання дискових масивів і автоматизації резервного копіювання. Проте використання фірмових апаратних платформ жорстко обмежує гнучкість інфраструктури, адже системний адміністратор повністю залежить від закритих оновлень безпеки виробника та життєвого циклу підтримки конкретного заліза. Будь-яке апаратне розширення, наприклад, модернізація центрального процесора, встановлення додаткових модулів оперативної пам'яті або інтеграція сторонніх плат абонентського розширення мережі, у таких системах є або фізично неможливими, або фінансово невиправданими через прив'язку до фірмових комплектуючих. Крім того, інтеграція закритих NAS-систем зі специфічними відкритими сервісами моніторингу метрик у реальному часі є суттєво обмеженою або потребує складних обхідних шляхів.

На противагу закритим системам, програмний комплекс з відкритим вихідним кодом Samba пропонує промисловий рівень надійності, повну свободу від ліцензійних платежів та можливість розгортання на будь-якому стандартному серверному обладнанні архітектури x86-64 [15] під керуванням операційної системи Linux. Для забезпечення наочності та швидкого оцінювання техніко-економічних параметрів існуючих рішень було сформовано спрощену порівняльну таблицю. Вона побудована за принципом прямого зіставлення ключових факторів, що дозволяє миттєво визначити переваги кожного підходу при першому перегляді.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

Таблиця 1.4

Спрощений порівняльний аналіз рішень для файлових сховищ

Критерій порівняння	Windows File Server	Апаратні NAS (Synology/QNAP)	Комплекс Samba (Linux)
Вартість ліцензій	Дуже висока (платна ОС + CAL)	Середня (включена у вартість заліза)	Безкоштовно (Open Source)
Залежність від вендора	Повна (ризик санкцій та блокувань)	Повна (обмежений цикл підтримки)	Відсутня (повна незалежність)
Апгрейд обладнання	Стандартний (будь-яке серверне залізо)	Обмежений або заблокований виробником	Необмежений (можливе будь-яке залізо)
Складність кастомізації	Висока (рамки ліцензії Microsoft)	Критична (закрита модульна ОС)	Мінімальна (повний доступ до коду)

Глибокий аналіз архітектури Samba показує, що цей комплекс не просто копіює функції файлового сервера, а повністю емулює поведінку підсистем Windows на рівні ядра мережевої взаємодії. Функціонування програмного комплексу Samba забезпечується завдяки чіткій взаємодії трьох основних системних служб операційної системи Linux, які працюють у фоновому режимі як демони [13]. Головним компонентом є демон `smbd` [11], який безпосередньо реалізує протокол спільного доступу SMBv3. Він обробляє поточкові запити читання й запису файлів, відповідає за автентифікацію користувачів під час підключення до мережеских папок, реалізує механізми блокування файлів для запобігання одночасному перезапису та транслює права доступу Windows ACL у системні дозволи POSIX ACL файлової системи Linux.

За зворотну сумісність із застарілими протоколами розрізнення імен, підтримку служби WINS та відображення сервера в класичному мережевому оточенні операційних систем минулих поколінь відповідає демон `nmbd`. У сучасних гетерогенних мережах, де розгортається автономна модель сховища (Standalone Server), ключову роль відіграє головний демон `smbd`. Замість використання інтеграційних містків на зразок служби `winbindd` для зв'язку з централізованими службами каталогів Active Directory, авторизаційний контур замикається безпосередньо на рівні локальної підсистеми Name Service Switch (NSS) операційної системи Linux та внутрішніх механізмів Samba. Взаємодія з користувачами та групами базується на локальних облікових записах, які зберігаються у файлах `/etc/passwd` та `/etc/group` на самому Linux-сервері, та дублюються

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

у високопродуктивній локальній базі даних криптографічних хешів паролів passwd.tdb. Це забезпечує повну автономність кожного вузла мережі, гарантує миттєву локальну валідацію токенів автентифікації без затримок на мережеві запити до сторонніх контролерів та виключає розсинхронізацію прав доступу суб'єктів.

Важливим технологічним аспектом при розгортанні Samba є вибір базової локальної файлової системи на сервері під керуванням Linux, оскільки саме вона безпосередньо взаємодіє з демоном smbd і визначає швидкість обробки операцій введення-виведення (I/O). Найбільш поширеними варіантами в інженерній практиці є ext4, XFS та Btrfs [14]. Кожна з них має унікальні особливості підтримки розширених атрибутів файлів (xattr), які Samba використовує для збереження дескрипторів безпеки Windows NT.

Таблиця 1.5

Порівняння локальних файлових систем Linux для інтеграції з Samba

Файлова система	Продуктивність з великими файлами	Робота з великою кількістю дрібних файлів	Підтримка розширених атрибутів (xattr)	Вбудовані технології знімків (Snapshots)
ext4	Середня/Висока	Дуже висока	Обмежена за розміром блоку атрибутів	Відсутні (потребує зовнішнього LVM)
XFS	Максимальна	Висока	Необмежена (ідеально для великих ACL)	Відсутні (потребує зовнішнього LVM)
Btrfs	Низька (через Copy-on-Write)	Середня	Повна	Вбудовані (підтримка Shadow Copy)

Зіставлення характеристик показує, що для корпоративного сховища з великим обсягом документів та складними вкладеними правами доступу найбільш раціональним є використання файлової системи XFS. Вона не має жорстких обмежень на розмір розширених атрибутів, що гарантує коректне збереження навіть дуже довгих списків доступу Windows ACL, а також демонструє стабільну швидкість при паралельній роботі багатьох користувачів.

Важливим етапом проектування масштабованих корпоративних сховищ є впровадження технології логічного розподілу простору імен DFS (Distributed File System). У контексті даної роботи під терміном DFS розуміється MSDFS —

реалізація простору імен DFS у складі програмного комплексу Samba, сумісна з клієнтами Microsoft Windows. Далі у тексті обидва терміни вживаються як рівнозначні. При побудові архітектури необхідно чітко розрізняти цю технологію від фізично розподілених файлових систем, таких як Ceph або GlusterFS [12]. Фізично розподілені системи призначені для об'єднання дискових масивів багатьох серверів в один спільний пул на низькому рівні. У таких системах кожен завантажений файл розбивається на дрібні блоки, які дублюються та розподіляються через мережу між різними фізичними носіями на різних серверах для забезпечення абсолютної апаратної відмовостійкості та захисту від виходу з ладу цілих стійок обладнання. Проте розгортання та супровід таких систем, як Ceph, вимагає значних обчислювальних ресурсів, наявності виділених швидкісних мереж зв'язку (від 10 GbE та вище) та є надлишковим для стандартного офісного документообігу організації, оскільки створює критичні накладні витрати на синхронізацію метаданих.

Натомість технологія DFS Namespaces працює виключно на логічному прикладному рівні, взагалі не змінюючи фізичну структуру зберігання даних на дисках і не виконуючи надлишкової реплікації бінарних блоків між серверами. Вона виконує роль інтелектуального мережевого покажчика, створюючи для користувачів єдине віртуальне дерево каталогів з однією спільною точкою входу. Коли користувач відкриває у провіднику віртуальну папку, служба DFS прозоро і непомітно для нього перенаправляє клієнтський запит на конкретний фізичний сервер Samba. Обґрунтування вибору DFS для масштабу організації базується на забезпеченні максимальної зручності користувачів, простоті додавання нових серверів без зміни мережових шляхів та мінімальних накладних витратах на трафік, оскільки подальший обмін файлами йде напрямку між клієнтом та цільовим сервером.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

Таблиця 1.6

**Аналіз розбіжностей між логічним (DFS) та фізичним (Ceph/GlusterFS)
розподілом даних**

Технічний критерій аналізу	Логічний розподіл (DFS Namespaces)	Фізичний розподіл (Ceph / GlusterFS)
Рівень абстракції системи	Логічний розподіл (DFS Namespaces)	Низькорівневий (робота блоками даних)
Фізичне переміщення файлів	Прикладний рівень (перенаправлення шляхів)	Постійне (файли фрагментуються та розподіляються по вузлах мережі)
Вимоги до швидкості мережі	Відсутнє (файли лежать цілісно на серверах)	Критичні (потребує 10/40 Gbps, низький Latency)
Накладні витрати на серверах	Стандартні (1 Gbps LAN повністю достатньо)	Високі (постійний прорахунок хешів та реплікацій)
Складність відновлення при збої	Мінімальні (короткий етап видачі посилання)	Надзвичайно висока (реконструкція всього кластера)

Впровадження DFS простору імен дозволяє вирішити ключову проблему великих підприємств — забезпечити безперервність доступу. Якщо один із серверів Samba потребує планового технічного обслуговування, адміністратор може скопіювати його вміст на резервний майданчик і просто змінити кінцеву точку в налаштуваннях DFS. Користувачі продовжуватимуть заходити за тією самою віртуальною адресою, навіть не підозрюючи, що в цей момент дані віддаються з іншого фізичного пристрою. Це мінімізує людський фактор, усуває необхідність переналаштування робочих станцій та знижує навантаження на службу технічної підтримки підприємства.

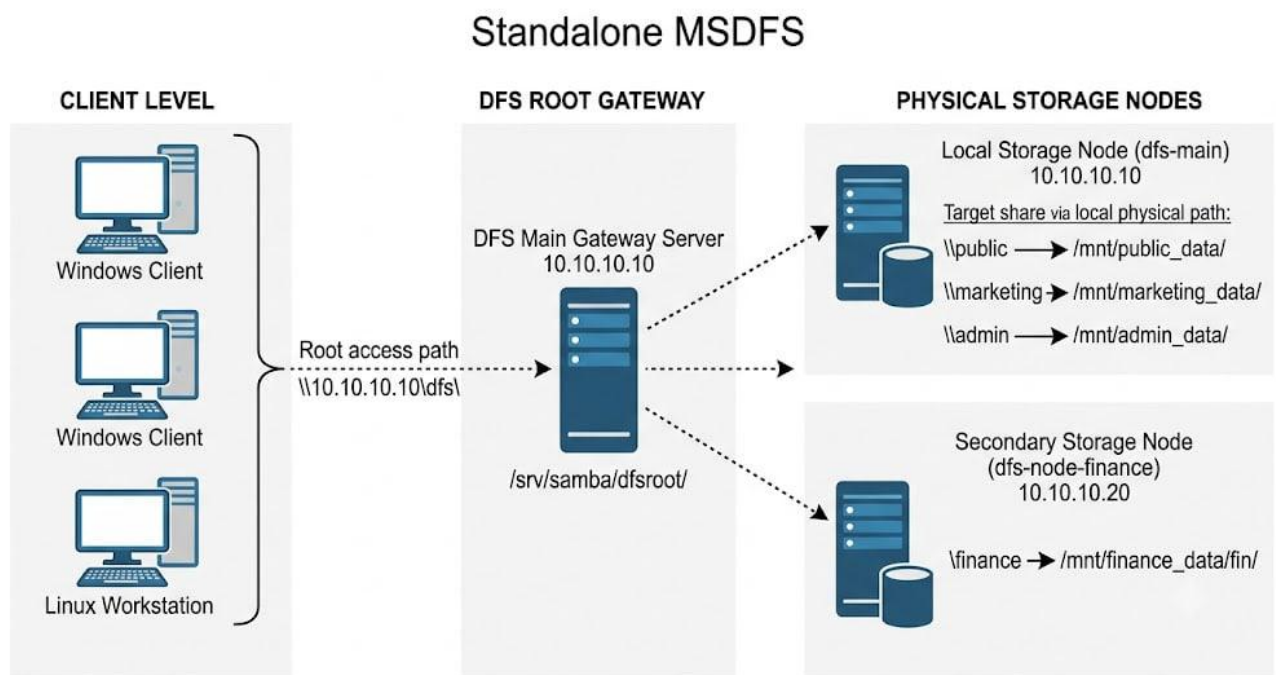


Рисунок 1.4 – Архітектура логічного об'єднання фізично розрізнених ресурсів за допомогою технології DFS

1.4 Огляд аналогічних рішень та існуючих реалізацій

Важливим етапом проектування будь-якої автоматизованої системи є проведення детального аналізу існуючих аналогів, розгорнутих у промислових умовах, а також вивчення споріднених науково-практичних праць з електронного репозиторію університету. Це дозволяє уникнути типових інженерних помилок, оцінити ефективність відомих конфігурацій та чітко виділити наукову новизну й практичну цінність власного проектного рішення. Під час аналізу було розглянуто кілька впроваджених архітектурних моделей, що використовуються для вирішення завдань централізованого зберігання корпоративного контенту на підприємствах різного масштабу.

Перша поширена реалізація, яка часто зустрічається в роботах репозиторію, базується на використанні комерційної операційної системи Windows Server з активованими ролями File and Storage Services та DFS Namespaces. У таких проектах відмовостійкість і балансування навантаження зазвичай реалізуються через службу реплікації DFSR (Distributed File System Replication), яка синхронізує файли між кількома серверами Microsoft у фоновому режимі. Головною перевагою

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

такої схеми є автоматизація процесу синхронізації та наявність зручних засобів моніторингу через системний журнал подій Windows Event Viewer.

Проте, як було зазначено в попередніх підрозділах, це рішення повністю прив'язане до пропрієтарної екосистеми, що створює критичні фінансові ризики для вітчизняних підприємств в умовах сучасного ринку. Крім того, аналіз показує, що служба DFSR при роботі з великими файлами (наприклад, образами дисків або базами даних) часто створює конфлікти версій і сильно перевантажує дискову підсистему серверів, викликаючи затримки в роботі користувачів.

Друга типова реалізація, що розглядається в академічних дослідженнях, описує побудову сховища на базі чистих Linux-рішень з використанням мережевої файлової системи NFS для зв'язку між серверами та клієнтами. У таких конфігураціях для забезпечення високої доступності даних застосовують технологію DRBD (Distributed Replicated Block Device), яка виконує поблокове дзеркалювання жорстких дисків через локальну мережу між двома серверами (схема Active-Passive кластера).

Така архітектура демонструє надзвичайно високу швидкість роботи та миттєве переключення на резервний вузол у разі апаратної аварії головного сервера. Але головне обмеження цього аналога полягає в орієнтованості виключно на Linux-клієнтів. Спроба підключити до такого кластера офісні комп'ютери під керуванням Windows вимагає розгортання додаткових шлюзів трансляції протоколів, що різко знижує загальну надійність системи, ламає кодування символів та ускладнює процес адміністрування прав доступу.

Третя група аналогічних рішень базується на використанні готових програмно-апаратних комплексів (TrueNAS Core, OpenMediaVault), які встановлюються на виділені сервери та використовують файлову систему ZFS. Ці рішення є дуже потужними, оскільки ZFS забезпечує вбудований контроль цілісності даних, миттєві знімки (Snapshots) та стиснення файлів "на льоту". Проте такі системи розгортаються як монолітні "чорні скриньки" із закритими внутрішніми механізмами керування через веб-інтерфейс, що робить неможливим гнучке налаштування розподіленого простору імен DFS поверх кількох окремих

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

машин та суттєво обмежує можливості низькорівневого моніторингу системних метрик за допомогою сторонніх легковажних скриптів.

Таблиця 1.7

Порівняльний аналіз архітектурних реалізацій файлових сховищ

Параметр порівняння	Комерційний аналог (Windows Server + DFSR)	UNIX-аналог (Linux + NFS + DRBD)	Монолітний NAS (TrueNAS / ZFS)	Розроблюване рішення (Samba + DFS + MQTT)
Ліцензійна незалежність	Відсутня (вища вартість)	Повна (Open Source)	Повна (Open Source)	Повна (Open Source)
Кросплатформність (Гетерогенність)	Повна	Обмежена (тільки Linux)	Середня	Повна (Windows + Linux)
Логічна централізація (DFS)	Вбудована	Відсутня	Відсутня	Вбудована (простір імен)
Гнучкість моніторингу метрик	Обмежена інструментами ОС	Потребує важких агентів	Обмежена веб-інтерфейсом	Максимальна (Python + MQTT)

Головна відмінність, перевага та інженерна новизна розроблюваного у даній роботі проекту, порівняно з усіма існуючими аналогами, полягає в інтеграції розподіленої файлової архітектури Samba DFS із превентивною системою автоматизованого моніторингу метрик у реальному часі на базі легковажного стеку Python, MQTT та Node-RED.

У більшості розглянутих аналогів та робіт з репозиторію контроль стану сховища є реактивним — адміністратор дізнається про проблему (наприклад, переповнення дисків на одному з прихованих вузлів DFS, критичний перегрів процесора або вичерпання RAM під час пікових завантажень великих файлів) лише після того, як сервер перестає відповідати на запити користувачів, що призводить до зупинки бізнес-процесів компанії. Крім того, класичні системи моніторингу (такі як Zabbix або Prometheus) потребують встановлення важких агентів супроводу, які самі по собі споживають значну частину обчислювальних ресурсів серверів.

У нашому проекті реалізовано кардинально інший підхід. На кожному серверному вузлі, який входить у структуру розподіленого простору імен DFS, розгортається мінімалістичний фоновий скрипт на базі мови Python та бібліотеки

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

psutil. Цей скрипт практично не створює навантаження на систему, зчитує апаратні показники серверів кожні 10 секунд [4] і миттєво відправляє їх у вигляді компактних JSON-повідомлень на центральний брокер за допомогою ультралегкого протоколу MQTT [1, 2] (порт 1883). Центральний сервер Node-RED приймає ці дані [3] та візуалізує їх на динамічному дашборді.

Така архітектура дозволяє реалізувати проактивне адміністрування: система візуально сигналізує про небезпечне зростання навантаження або дефіцит вільного місця на конкретному сервері Samba задовго до того, як це вплине на доступність віртуального простору імен DFS для кінцевих користувачів. Це забезпечує унікальний баланс між високою продуктивністю зберігання даних, абсолютною безкоштовністю ПЗ та гарантованою відмовостійкістю всієї корпоративної інфраструктури.

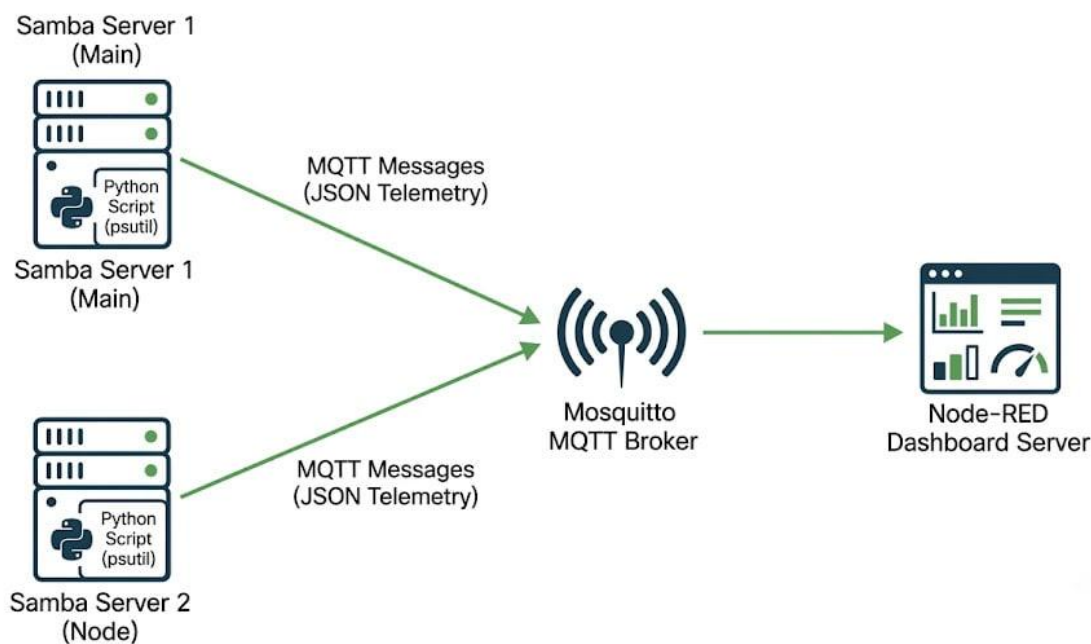


Рисунок 1.5 – Архітектура інтегрованого моніторингу розподіленого сховища

1.5 Постановка задачі та вимоги до системи

Фінальним етапом аналізу предметної області та вивчення існуючих аналогів є чітке формулювання технічного завдання, а також визначення повного переліку вимог до розроблюваної системи. Постановка задачі базується на необхідності усунення всіх деструктивних факторів децентралізованого зберігання даних, які

були розглянуті у першому підрозділі, із одночасним забезпеченням високої доступності та проактивного контролю за станом обладнання. Для побудови ефективного інженерного рішення всі вимоги до файлового сховища поділяються на дві фундаментальні категорії: функціональні, які визначають безпосередні обов'язки та можливості системи, та нефункціональні, що описують якісні характеристики, обмеження та умови експлуатації платформи.

Функціональні вимоги до проєктованого сховища визначають архітектурні можливості взаємодії користувачів та адміністраторів із ресурсами системи. По-перше, система повинна забезпечувати створення, конфігурування та ізоляцію спільних мережевих папок для різних структурних підрозділів організації, таких як бухгалтерія, юридичний департамент, відділ маркетингу та загальний публічний каталог. По-друге, обов'язковою є реалізація гнучкої моделі розмежування прав доступу на основі користувачів та груп, що дозволяє призначати права на читання, запис або повну модифікацію контенту для конкретних відділів, виключаючи несанкціонований доступ суміжних служб. По-третє, архітектура повинна реалізовувати єдиний простір імен за допомогою технології DFS, щоб кінцеві користувачі мали єдину віртуальну точку входу і не залежали від зміни фізичних серверів чи IP-адрес. Нарешті, система має підтримувати прозору інтеграцію з корпоративними обліковими записами і забезпечувати автоматизоване резервне копіювання критично важливих даних за розкладом, а також превентивний збір та візуалізацію апаратних метрик (навантаження на процесор, пам'ять, диски) з кожного вузла DFS.

Нефункціональні вимоги фокусуються на якісних параметрах роботи платформи та критеріях її надійності. У контексті продуктивності система повинна демонструвати мінімальну затримку (таймаут) при первинному зверненні клієнта до віртуального кореня DFS та забезпечувати максимальну пропускну здатність, близьку до лімітів гігабітної локальної мережі, при копіюванні великих масивів даних. Вимоги до безпеки включають обов'язкову автентифікацію для всіх ізольованих мережевих ресурсів (гостьовий доступ дозволяється лише для загальнодоступних ресурсів [public] та кореня простору імен [dfs]), криптографічне

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

шифрування службового трафіку за допомогою протоколу SMBv3, ізоляцію контурів адміністрування та захист від несанкціонованої модифікації файлів конфігурації. У розрізі керованості та масштабованості система має бути спроектована так, щоб системний адміністратор міг оперативно додавати нові фізичні сервери Samba або вилучати застарілі вузли без зупинки роботи всього простору імен сховища та без необхідності переналаштування робочих станцій користувачів.

Таблиця 1.8

Спрощена матриця функціональних та нефункціональних вимог до системи

Тип вимоги до системи	Конкретний технічний прояв вимоги	Цільовий інженерний результат
Функціональна	Створення ізольованих мережових папок	Повна структуризація файлів по відділах
	Логічне об'єднання через DFS Namespaces	Наявність однієї спільної точки входу
	Інтеграція прав доступу та користувачів	Захист конфіденційних даних від суміжних служб
	Автоматичний збір метрик через MQTT	Превентивне виявлення перевантажень заліза
Нефункціональна	Висока продуктивність у LAN мережі	Швидкість передачі даних на рівні ємності каналу
	Криптографічний захист зв'язку	Шифрування потоку даних за стандартом SMBv3
	Модульна керованість інфраструктури	Швидкий апгрейд сховища без простою сервісів

На основі сформованих вимог постановка задачі для даної кваліфікаційної роботи полягає у розробці та практичній реалізації комплексної архітектурної моделі розподіленого файлового сховища. Практична реалізація передбачає використання операційної системи Ubuntu Server як стабільної та безкоштовної основи, розгортання та тонке конфігурування серверів Samba для сумісності з гетерогенним середовищем, налаштування логічної структури простору імен DFS, створення гнучких правил безпеки на базі POSIX/Windows ACL, а також написання та інтеграцію легковажної підсистеми збору телеметрії через Python/MQTT з виводом графіків на панель Node-RED.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колощук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

Висновки до розділу 1

У першому розділі було виконано комплексний аналіз предметної області та обґрунтовано актуальність модернізації інфраструктури зберігання даних на підприємствах. На основі детального вивчення децентралізованих систем виявлено ключові деструктивні фактори, такі як розрізненість ресурсів, неконтрольоване дублювання файлів, критична складність ручного керування правами доступу, відсутність єдиної точки входу та проблеми з організацією регулярного резервного копіювання. Завдяки аналізу мережевих протоколів доведено технологічну доцільність вибору протоколу SMBv3 як єдиного відкритого стандарту, здатного забезпечити прозору кросплатформну взаємодію в гетерогенному середовищі без встановлення додаткового клієнтського ПЗ на робочі станції під управлінням Windows та Linux.

Проведено порівняльний аналіз комерційних та відкритих технологічних рішень, за результатами якого обґрунтовано вибір програмного комплексу Samba під керуванням операційної системи Linux, що дозволяє повністю ліквідувати фінансову залежність від пропрієтарних ліцензій Microsoft та закритих рішень NAS-вендорів. Чітко розмежовано поняття логічно розподілених просторів імен DFS та фізично розподілених систем типу Ceph, що дозволило довести раціональність використання технології DFS Namespaces для стандартного документообігу організації з огляду на мінімальні накладні витрати на трафік. Огляд існуючих аналогів дозволив виділити головну перевагу та інженерну новизну проекту — інтеграцію сховища з превентивною системою автоматизованого моніторингу метрик серверів у реальному часі на базі легковажного стеку Python та протоколу MQTT. Сформовано повну матрицю функціональних і нефункціональних вимог та визначено постановку задачі на проектування, що виступає базою для наступного розділу роботи.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

РОЗДІЛ 2. МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ ФАЙЛОВОГО СХОВИЩА

2.1 Розробка архітектури системи

Проектування архітектури розподіленого файлового сховища є визначальним етапом роботи, оскільки створена модель повинна забезпечувати безперебійну взаємодію між різнорідними клієнтськими операційними системами, логічно об'єднувати фізичні накопичувачі та гарантувати постійний превентивний контроль за станом апаратних ресурсів. Загальна архітектурна модель розроблюваного рішення базується на трирівневій концепції абстракції даних, де чітко розмежовано рівень кінцевих споживачів (гетерогенні клієнти), рівень логічної маршрутизації (DFS (MSDFS)-простір імен) та рівень безпосереднього фізичного зберігання (сервери Samba). Така побудова дозволяє ізолювати фізичну структуру серверного заліза від логічного представлення каталогів для користувачів, що надає системі максимальної гнучкості під час експлуатації та подальшого розширення.

На першому рівні загальної архітектурної моделі знаходяться клієнтські робочі станції підприємства. Особливістю проєктованого середовища є його гетерогенність, тобто одночасна присутність у локальній мережі комп'ютерів під керуванням операційних систем Microsoft Windows та різноманітних дистрибутивів GNU/Linux. Клієнтські машини взаємодіють із системою виключно через стандартні системні виклики протоколу SMBv3 (порт 445). Важливою інженерною перевагою такої моделі є те, що користувачі не мають прямого доступу до консолі серверів або до їхніх локальних файлових систем. Для кінцевого працівника вся взаємодія зводиться до роботи у звичному графічному інтерфейсі системного файлового менеджера, де віддалені мережеві ресурси відображаються як стандартні локальні каталоги.

Другий рівень архітектури є центральним координаційним вузлом і реалізує технологію логічного розподілу простору імен DFS Namespaces. Цей рівень розгорнутий на базі головного сервера сховища і виступає в ролі інтелектуального комутатора мережевих шляхів. Коли клієнтська станція ініціює підключення до

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

єдиної точки входу, цей рівень обробляє запит і повертає клієнту так зване реферал-посилання (Referral) [7], яке містить точну інформацію про те, на якому саме фізичному сервері Samba зберігається потрібна папка. Оскільки цей процес відбувається автоматично на рівні протоколу SMB, користувач бачить суцільне дерево папок і не помічає перенаправлення трафіку, що повністю вирішує проблему плутанини в мережевих шляхах та IP-адресах.

Третій рівень представлений безпосередньо фізичними серверами Samba, які здійснюють безпосереднє збереження інформації на своїх дискових масивах під керуванням операційної системи Ubuntu Server. У межах даного проекту цей рівень має розподілену структуру та складається з двох виділених вузлів: головного сервера сховища та спеціалізованого сервера фінансового департаменту. Кожен із цих вузлів працює незалежно, має власну локальну файлову систему XFS та обслуговує свій унікальний набір мережевих папок.

Таблиця 2.1

Функціональний розподіл та параметри архітектурних вузлів сховища

Назва серверного вузла	Роль в архітектурі DFS	Обслуговувані мережеві ресурси	Операційна система та сервіси
Головний сервер сховища	Кореневий вузол простору імен DFS, фізичний сервер	Віртуальний корінь DFS, загальні папки, публічний каталог	Ubuntu Server, Samba (smbd, nmbd), Mosquitto MQTT Broker
Фінансовий сервер сховища	Підпорядкований цільовий вузол (DFS Link)	Закриті мережеві папки бухгалтерії та фінансового відділу	Ubuntu Server, Samba (smbd), Python telemetry agent

Структурна схема рішення деталізує взаємозв'язки між апаратними та програмними компонентами системи, а також описує напрямки руху інформаційних потоків і службового трафіку телеметрії. На відміну від класичних децентралізованих схем, де кожен клієнт підключався до серверів хаотично, у розробленій структурній схемі чітко виділено єдиний вектор взаємодії. Клієнти звертаються до віртуального імені домену, а підсистема DFS автоматично розподіляє навантаження, надсилаючи запити на відповідний сервер Samba. Паралельно з цим, у структуру рішення інтегровано незалежний контур збору

апаратних метрик, який працює у фоновому режимі та не впливає на швидкість передачі файлів користувачами.

Для моніторингу стану заліза на кожному з двох серверів Samba функціонує легковажний агент на базі мови Python, який кожні 10 секунд зчитує показники утилізації процесора, оперативної пам'яті та дискового простору. Ці дані упаковуються в компактний текстовий формат JSON і передаються через внутрішню мережу на порт 1883 головного сервера, де розгорнуто брокер повідомлень Mosquitto MQTT. Брокер здійснює миттєву маршрутизацію отриманих пакетів телеметрії на сервер візуалізації Node-RED, який обробляє вхідні потоки даних та відображає їх на інтерактивному графічному дашборді адміністратора. Така структура гарантує проактивний контроль за кожним елементом розподіленої системи. Для систематизації параметрів взаємодії між компонентами структурної схеми сформовано спрощену матрицю інтерфейсів взаємодії.

Таблиця 2.2

Матриця мережевих інтерфейсів та протоколів структурної схеми

Напрямок взаємодії компонентів	Використовуваний протокол	Мережевий порт	Функціональне призначення потоку
Клієнт — Простір імен DFS	SMBv3	TCP 445	Отримання реферал-посилань, навігація
Клієнт — Сервери Samba	SMBv3	TCP 445	Читання, запис та модифікація файлів
Сервери Samba — MQTT Брокер	MQTT (JSON)	TCP 1883	Передача апаратних метрик телеметрії
MQTT Брокер — Сервер Node-RED	MQTT (JSON)	TCP 1883	Перенаправлення даних для аналітики
Адміністратор — Node-RED Дашборд	HTTP / HTTPS	TCP 1880	Візуальний моніторинг стану інфраструктури

Описана структурна архітектура дозволяє повністю нівелювати ризики, пов'язані з відмовою окремих елементів. Оскільки логіка керування зосереджена на рівні конфігураційних файлів Samba та простору імен DFS, додавання третього або четвертого сервера у фінансовий чи будь-який інший департамент вимагатиме лише внесення одного рядка в конфігурацію віртуального дерева каталогів та запуску копії Python-скрипта моніторингу на новому залізі. Це доводить високу

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

масштабованість, надійність та експлуатаційну ефективність розробленої структурної моделі.

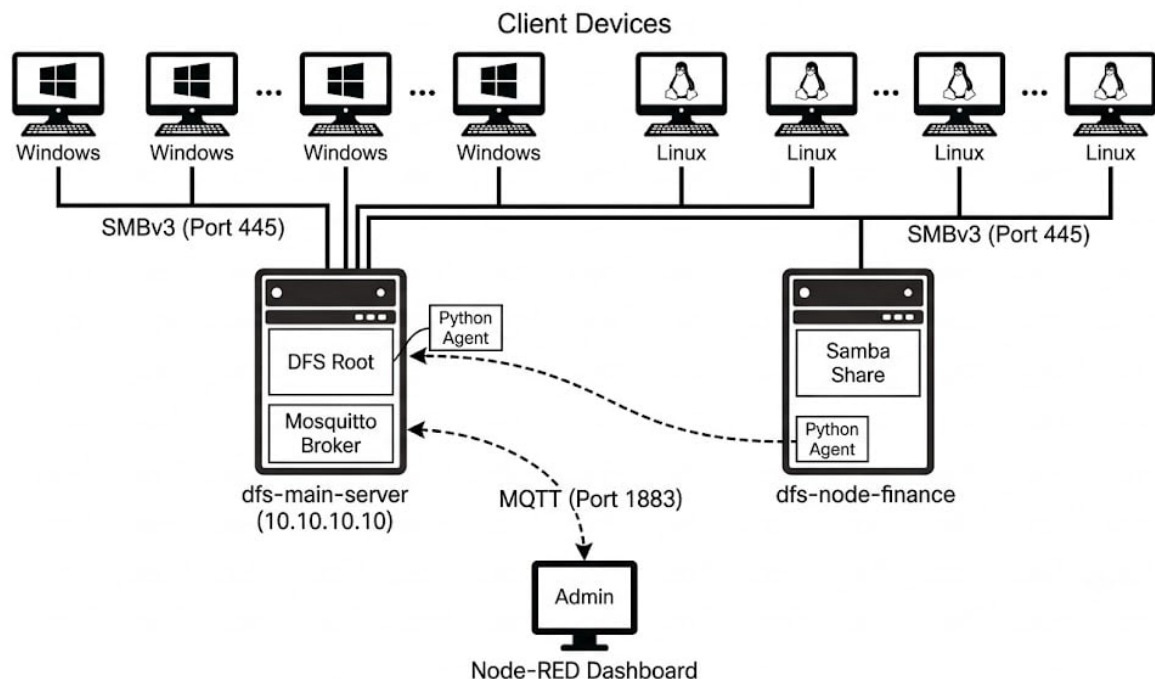


Рисунок 2.1 – Структурна схема архітектури розподіленого сховища Samba DFS та інтегрованої підсистеми моніторингу

2.2 Моделювання мережевої інфраструктури організації

Проектування розподіленого файлового сховища вимагає детального моделювання мережевої інфраструктури підприємства, оскільки стабільність логічного простору імен DFS безпосередньо залежить від архітектурної топології мережі, коректності побудови планів адресації та надійності каналів зв'язку. У межах цього проєкту мережева інфраструктура моделюється як структурована локальна обчислювальна мережа (LAN), сегментована за допомогою технології віртуальних локальних мереж (VLAN). Такий підхід дозволяє ізолювати критично важливі серверні вузли Samba від загального користувацького трафіку, мінімізувати широкомовний шторм та забезпечити високий рівень безпеки інформаційних потоків на рівні комутації.

Базова топологія мережі базується на ієрархічній моделі, де чітко виділено ядро мережі, рівень розподілу та рівень доступу. Серверні компоненти сховища розміщуються у спеціально виділеному сегменті даних — серверній демілітаризованій зоні (DMZ). Головний сервер сховища з адресою 10.10.10.10 та

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				38
Змн.	Арк.	№ докум.	Підпис	Дата		

додатковий фінансовий вузол з адресою 10.10.10.20 підключаються безпосередньо до гігабітних портів комутатора ядра. Робочі станції користувачів (бухгалтерія, маркетинг, юридичний відділ) знаходяться в окремих клієнтських підмережах і взаємодіють із серверами через маршрутизатор, на якому налаштовані правила міжсегментного екранування. Для чіткої структуризації та превентивного виключення конфліктів IP-адрес у межах проєкту розроблено єдину схему адресації інфраструктури.

Таблиця 2.3

Схема мережевої адресації та сегментації інфраструктури сховища

Назва мережевого сегмента / вузла	Ідентифікатор VLAN	Діапазон IP-адрес / IP-адреса	Маска підмережі	Інфраструктурне призначення
Server DMZ (Сегмент серверів)	VLAN 10	10.10.10.0 / 24	255.255.255.0	Розміщення серверів Samba, MQTT та Node-RED
dfs-main-server (Головний)	VLAN 10	10.10.10.10	Host (/32)	Корінь DFS, загальні папки, Mosquitto Broker
dfs-node-finance (Фінанси)	VLAN 10	10.10.10.20	Host (/32)	Цільовий вузол DFS, папки фінансового відділу
Corporate Clients (Клієнти)	VLAN 20	10.10.20.0 / 24	255.255.255.0	Робочі станції користувачів Windows/Linux
Management Network (Управління)	VLAN 99	10.10.99.0 / 24	255.255.255.0	Адміністративний контур для SSH та Node-RED GUI

Для детального опису динаміки взаємодії між компонентами системи на етапі моделювання застосовується мова уніфікованого моделювання UML, зокрема діаграма послідовності (Sequence Diagram). Вона дозволяє покроково відстежити життєвий цикл запиту клієнта при спробі отримати доступ до файлів через віртуальний простір імен DFS. Сценарій взаємодії починається з того, що користувач ініціює запит на підключення до спільного простору. Клієнтська операційна система відправляє первинний SMB-запит на головний сервер сховища, який є коренем DFS. Головний сервер аналізує запит, звіряє його зі своєю внутрішньою картою простору імен і повертає клієнту спеціальну структуру — DFS Referral, що містить IP-адресу цільового сервера, де фізично розташовані дані.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

Отримавши це посилання, клієнтський комп'ютер автоматично і прозоро для користувача перенаправляє потоковий трафік безпосередньо на другий сервер Samba, встановлюючи з ним пряму сесію для читання та запису файлів. Паралельно з цим процесом на обох серверах безперервно функціонує контур збору телеметрії. Фоновий Python-агент на фінансовому сервері зчитує показники заліза та через протокол MQTT відправляє JSON-пакет на брокер Mosquitto, розгорнутий на головному сервері. Брокер миттєво транслює повідомлення на сервер Node-RED, який оновлює дашборд адміністратора. Цей комплексний процес синхронізації логічних та сервісних потоків деталізовано у спрощеній матриці послідовності операцій.

Таблиця 2.4

Послідовність виконання операцій при доступі до DFS та передачі метрик

Номер кроку	Компонент-ініціатор запити	Компонент-отримувач запити	Використовувана операція / Протокол	Кінцевий результат операції
1	Робоча станція користувача	<i>dfs-main-server</i> (Кореневий)	Первинний запит шляху / SMBv3	Запит карти маршрутизації DFS
2	<i>dfs-main-server</i> (Кореневий)	Робоча станція користувача	Видача DFS Referral / SMBv3	Передача клієнту адреси цільового вузла
3	Робоча станція користувача	<i>dfs-node-finance</i> (Цільовий)	Пряме підключення до Share / SMBv3	Відкриття папки, робота з файлами
4	Python-агент (<i>dfs-node-finance</i>)	Mosquitto Broker (<i>dfs-main</i>)	Публікація метрик (Publish) / MQTT	Передача JSON-пакета з даними заліза
5	Mosquitto Broker (<i>dfs-main</i>)	Node-RED Dashboard Server	Трансляція даних (Subscribe) / MQTT	Візуалізація стану вузла на графіках

Моделювання такої двоконтурної мережевої взаємодії (де контур даних користувачів повністю відокремлений від контуру збору службової телеметрії) дозволяє гарантувати високу стійкість системи до пікових навантажень. Навіть якщо користувачі почнуть одночасно копіювати великі масиви проектної документації, забиваючи смугу пропускання протоколу SMB, легковажні MQTT-пакети моніторингу через свій мінімальний розмір безперешкодно доходитимуть до сервера Node-RED, забезпечуючи адміністратора актуальною інформацією про стан завантаження дисків та процесорів у реальному часі.

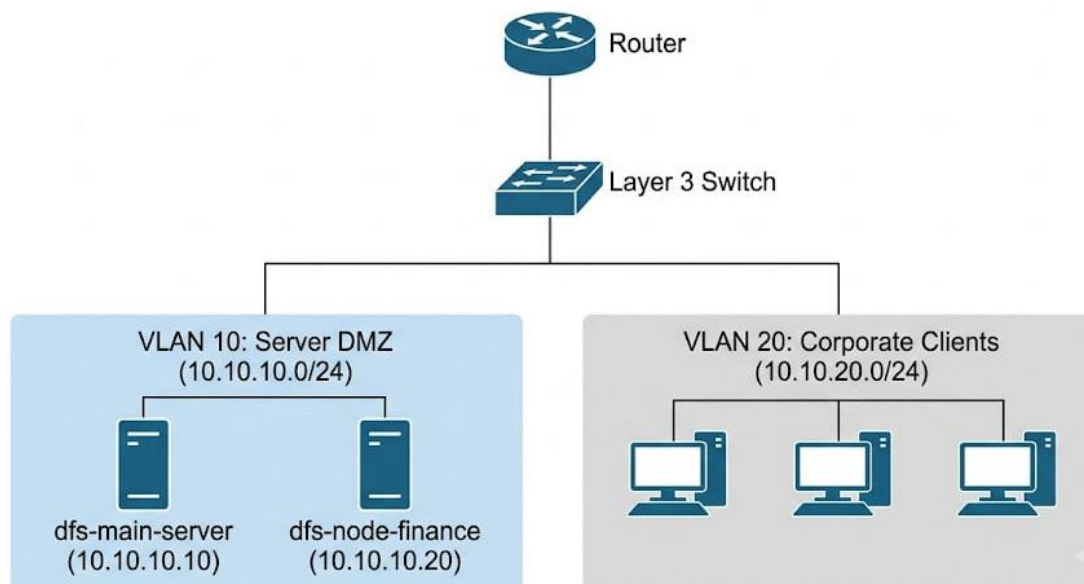


Рисунок 2.2 – Схема мережевої топології та сегментації підмереж файлового сховища

2.3 Проектування структури файлового сховища та DFS

Важливим етапом побудови розподіленої інфраструктури є детальне проектування логічної ієрархії каталогів та формування структури простору імен розподіленої файлової системи. На цьому етапі абстрактні архітектурні шари трансформуються в конкретну, зрозумілу для персоналу систему каталогів із чітким розмежуванням повноважень між структурними підрозділами компанії. Правильно спроектована ієрархія спільних ресурсів (**Shares**) дозволяє ліквідувати проблему неконтрольованого дублювання документів та забезпечити превентивний захист конфіденційної інформації від несанкціонованого доступу з боку суміжних відділів.

Логічна ієрархія сховища базується на принципі суворої відповідності структурі підприємства та розподіляється між двома наявними фізичними серверами. На головному сервері **dfs-main-server** розгортається загальнодоступний публічний каталог для обміну файлами між усіма працівниками, а також ізольовані папки адміністративного та маркетингового департаментів. На додатковому виділеному сервері **dfs-node-finance** зосереджуються закриті ресурси фінансового відділу та бухгалтерії, які вимагають підвищеного рівня ізоляції та окремих політик резервного копіювання.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

Таблиця 2.5

Структура та фізичне розміщення спільних мережесих папок (Shares)

Назва ресурсу (Share Name)	Абсолютний локальний шлях (Path)	Фізичний сервер (Host IP)	Тип даних та функціональне призначення каталогу
[dfs]	/srv/samba/dfsroot	10.10.10.10	Віртуальний корінь простору імен DFS (Точка входу)
[public]	/mnt/public_data	10.10.10.10	Загальний спільний ресурс для всіх співробітників компанії
[marketing]	/mnt/marketing_data	10.10.10.10	Ізольована робоча область відділу маркетингу та реклами
[admin]	/mnt/admin_data	10.10.10.10	Захищений системний том для потреб ІТ-адміністраторів
[finance]	/mnt/finance_data/fin	10.10.10.20	Обмежений конфіденційний ресурс фінансового підрозділу

Проектування простору імен DFS namespaces полягає у зведенні цих фізично розрізнених мережесих папок в єдину, монолітну віртуальну структуру. Простір імен створюється у вигляді віртуального дерева каталогів під керуванням головного сервера, який виступає коренем DFS (*DFS Root*). Замість того, щоб підключати окремо кожен IP-адресу, користувачі отримують доступ до єдиного мережесого шляху `\\company.local\dfs`, усередині якого опубліковані віртуальні посилання (*DFS Links*) на реальні папки серверів Samba.

Коли клієнт звертається до віртуального дерева, корінь DFS віддає йому реферал-посилання на конкретну цільову машину. Наприклад, перехід до папки `\\company.local\dfs\public` перенаправляє трафік на `\\10.10.10.10\public`, а клієнтський запит до папки `\\company.local\dfs\finance` автоматично, без участі користувача, транслюється на фізичний диск другого сервера — `\\10.10.10.20\finance`.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

Матриця трансляції віртуальних шляхів DFS у фізичні мережеві адреси

Віртуальний мережевий шлях у DFS	Фізичний UNC-шлях Samba	Кінцева IP-адреса вузла	Тип DFS компонента
\\company.local\dfs\	\\dfs-main-server\dfsroot	10.10.10.10	DFS Root (Корінь)
\\company.local\dfs\public	\\10.10.10.10\public	10.10.10.10	DFS Link (Посилання)
\\company.local\dfs\marketing	\\10.10.10.10\marketing	10.10.10.10	DFS Link (Посилання)
\\company.local\dfs\admin	\\10.10.10.10\admin	10.10.10.10	DFS Link (Посилання)
\\company.local\dfs\finance	\\10.10.10.20\finance	10.10.10.20	DFS Link (Посилання)
\\company.local\dfs\accounting	\\10.10.10.20\accounting	10.10.10.20	DFS Link (Посилання)

Така схема проєктування дозволяє повністю приховати від кінцевого користувача внутрішню топологію мережі та реальні IP-адреси серверів. Якщо в майбутньому обсяг фінансових документів перевищить місткість дисків сервера *dfs-node-finance*, адміністратор зможе розгорнути третій сервер, перенести туди папку *accounting* і просто оновити один рядок у конфігурації DFS посилань на головному вузлі. Для користувачів мережевий шлях *\\company.local\dfs\accounting* залишиться незмінним, що виключає необхідність перепідключення мережевих дисків на робочих станціях та забезпечує абсолютну гнучкість і масштабованість корпоративної інфраструктури.

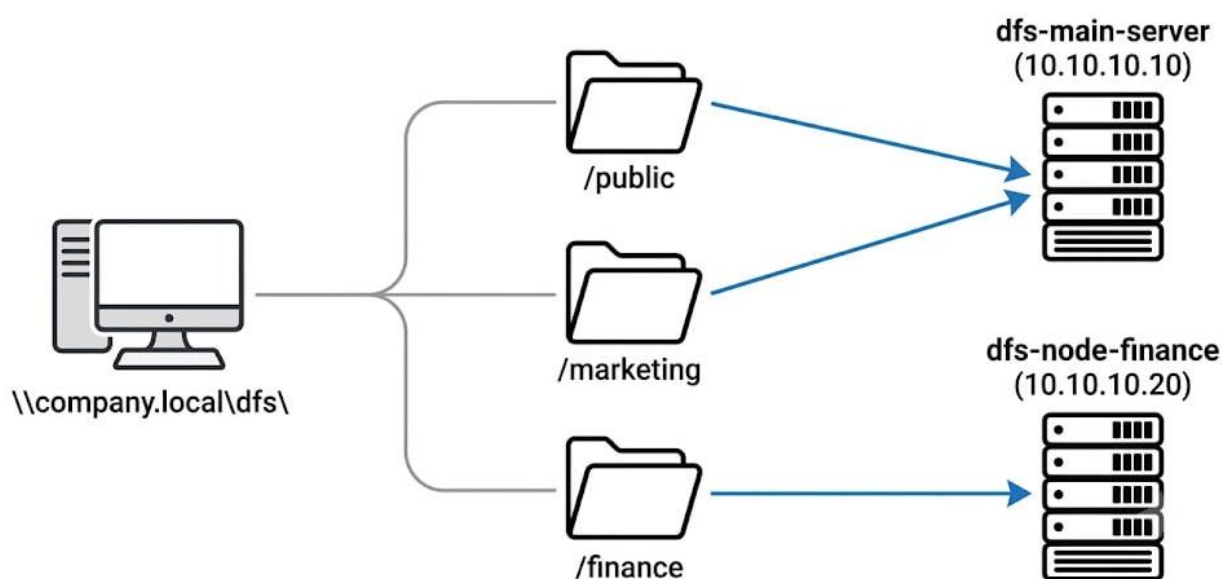


Рисунок 2.3 – Схема логічної структури та маршрутизації простору імен DFS Namespaces

2.4 Проектування моделі розмежування прав доступу та автентифікації

Забезпечення конфіденційності, цілісності та доступності інформаційних активів у межах розподіленої файлової інфраструктури потребує проектування чіткої та суворої моделі розмежування прав. При побудові архітектури корпоративного сховища було обрано автономну модель безпеки типу Standalone Server з використанням прикладного протоколу Samba. Відмова від доменної структури Active Directory та централізованих служб автентифікації на базі протоколу **Kerberos** чи демона **winbindd** дозволяє суттєво знизити накладні витрати на адміністрування мережі, усунути ризики виникнення єдиної точки відмови (Single Point of Failure) та оптимізувати швидкість обробки транзакцій введення-виведення у невеликих корпоративних середовищах.

Ідентифікація та автентифікація клієнтських робочих станцій під керуванням операційних систем сімейства Windows реалізується безпосередньо на рівні мережесокетів кожного окремого серверного вузла. Для збереження облікових записів та криптографічних хешів паролів Samba використовує вбудовану високопродуктивну локальну базу даних Trivial Database (TDB), яка активується в системних конфігураціях за допомогою директиви **passdb backend = tdbsam**. Авторизаційний контур побудовано на базі рольової моделі доступу (RBAC), де права суб'єктів визначаються їх належністю до конкретних локальних облікових записів операційної системи Linux, які попередньо синхронізовані з базою даних Samba за допомогою утиліти **smbpasswd**.

Для досягнення максимального рівня захищеності у розподіленій інфраструктурі спроектовано двоконтурну модель фільтрації доступу клієнтів. Перший контур безпеки діє на мережевому рівні прикладного протоколу SMBv3. За допомогою конфігураційного параметра **valid users** у файлі **smb.conf** система чітко обмежує коло осіб, які мають право ініціювати сесію підключення до конкретного мережевого ресурсу. Будь-які спроби неавторизованого, анонімного або гостьового підключення до ізольованих папок автоматично відхиляються на етапі SMB-рукошлякування (handshake). Другий контур безпеки функціонує безпосередньо на рівні локальної файлової системи Linux (Ext4/XFS) за допомогою механізму

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

розширених рекурсивних списків контролю доступу POSIX ACL [10]. Це дозволяє гнучко призначати права читання, запису та виконання (rwx) на рівні конкретних inode для локальних користувачів, повністю перекриваючи базові обмеження стандартної Unix-моделі прав доступу (UGO).

Таблиця 2.7

Класифікація суб'єктів, груп та функціональних ролей системи

Назва локальної групи безпеки Linux	Приклад облікового запису	Функціональна роль у системі	Об'єкт доступу (Директорія)
sysadmins	dima_admin	Повний адміністратор інфраструктури	Усі каталоги та конфігурації
finance	vova_fin	Керуючий фінансовими потоками	/dfs/finance
accounting	buh_olga	Операційний бухгалтер	/dfs/accounting
marketing	mark_devis	Спеціаліст із маркетингу та реклами	/dfs/marketing
Users	Усі користувачі	Базовий співробітник організації	/dfs/public

Модель розмежування прав доступу в розроблюваному сховищі є двоконтурною та реалізується на двох незалежних рівнях: на рівні мережевих папок (Share Permissions) та на рівні безпосередньої локальної файлової системи (File System Permissions). Для того щоб користувач зміг виконати будь-яку операцію з файлом (наприклад, видалити або модифікувати договір), його запит повинен успішно пройти перевірку на обох рівнях безпеки. Якщо на рівні мережевої папки Samba дозволено повний доступ, а на рівні локальної файлової системи Linux XFS встановлено обмеження «тільки для читання», підсистема безпеки заблокує спробу запису, керуючись принципом найбільш суворого обмеження.

На першому рівні (Share Permissions) конфігурування здійснюється безпосередньо в конфігураційному файлі smb.conf для кожного окремого ресурсу. Тут визначаються глобальні параметри доступності папки в мережі, дозволені локальні користувачі та переліки груп, які мають право ініціювати SMB-сесію. На другому рівні (File System Permissions) використовуються розширені списки контролю доступу (POSIX ACL) операційної системи Linux. Це дозволяє гнучко налаштовувати права на рівні окремих вкладених папок та файлів суб'єктів,

підтримуючи специфічні атрибути успадкування прав від батьківських каталогів до дочірніх об'єктів без залучення сторонніх служб ідентифікації.

Таблиця 2.8

Матриця розмежування прав доступу на рівнях Shares та POSIX ACL

Віртуальний каталог DFS	Права на рівні папки Samba (Share)	Права на рівні файлової системи (POSIX ACL)	Фінальний рівень прав для групи
/dfs/public	Read/Write для всіх	rwXrwxr-x (Група має повний доступ)	Читання та запис (видалення заборонено)
/dfs/marketing	Read/Write для marketing	rwXrwx--- (Лише для маркетингу)	Повний доступ для членів групи
/dfs/finance	Read/Write для finance	rwXrwx--- (Лише для фінансів)	Повний доступ для членів групи
/dfs/accounting	Read/Write для accounting	rwXrwx--- (Лише для бухгалтерії)	Повний доступ для членів групи
/dfs/admin	Read/Write для sysadmins	rwXrwx--- (Лише для адміністраторів)	Повний доступ (керування правами)

Така дворівнева архітектура безпеки гарантує абсолютну ізоляцію критичних корпоративних даних. Навіть якщо рядовий користувач спробує вручну підібрати шлях чи IP-адресу фінансового сервера *dfs-node-finance* в обхід віртуального дерева DFS, демон *smbd* відхилить його запит ще на етапі перевірки дескриптора безпеки мережевої папки, зафіксувавши спробу несанкціонованого доступу в системному лог-файлі. Це забезпечує надійний захист комерційної таємниці та повністю відповідає сучасним стандартам побудови захищених автономних корпоративних систем [22] зберігання даних.

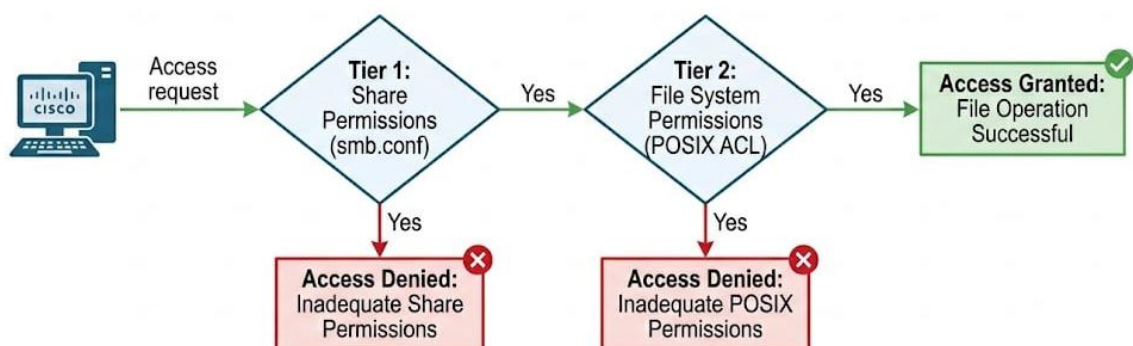


Рисунок 2.4 – Алгоритм дворівневої перевірки прав доступу в Samba

2.5 Моделювання підсистеми резервного копіювання

Проектування надійної підсистеми резервного копіювання є критично важливим етапом побудови розподіленого файлового сховища, оскільки централізація корпоративних даних в єдиному просторі імен DFS значно підвищує ризики їхньої одночасної втрати внаслідок апаратних збоїв, помилок персоналу або кібератак. За відсутності ефективної стратегії відновлення, будь-яка аварія на одному з вузлів Samba може призвести до незворотної втрати інтелектуальної власності підприємства та повної зупинки бізнес-процесів. Моделювання підсистеми резервування в цій роботі базується на впровадженні комплексної локальної стратегії, яка охоплює всі ключові компоненти розподіленої інфраструктури, включаючи критичні дані користувачів та системні конфігураційні файли.

Стратегія резервного копіювання розробляється з урахуванням специфіки гетерогенного середовища та вимог до мінімізації часу відновлення (RTO) і допустимої втрати даних (RPO). Основний акцент робиться на забезпеченні відмовостійкості критично важливих фінансових та адміністративних даних, розміщених на серверах сховища. Модель передбачає створення щоденних повних архівів (Full Backup) у моменти найменшої утилізації апаратних ресурсів. В якості основного інструментарію резервування обрано класичне поєднання системних утиліт tar та gzip, що дозволяє виконувати ефективне послідовне стиснення даних, гарантує монолітність результуючих файлів для запобігання пошкодженню структури каталогів, економить дисковий простір та виключає мережевий оверхед. Для систематизації параметрів розробленої стратегії сформовано спрощену матрицю резервного копіювання.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

Таблиця 2.9

Параметри резервування критичних даних фінансового підрозділу

Об'єкт резервування	Тип резервування (Type)	Назва файлу архіву (Format)	Глибина зберігання (Retention)	Цільовий каталог (Destination) та Логування
Фінансові дані (/mnt/finance_data/fin)	Повний (tar - czf)	fin-backup-YYYY-MM-DD-HHMMSS.tar.gz	7 днів (-mtime +7)	/home/sysadmin/backups (Лог: backup.log)

Модель передбачає використання архітектурного принципу локальної ізоляції та циклічної ротації резервних копій у межах виділеного адміністративного простору. Резервні копії критичних даних фінансового відділу з сервера dfs-node-finance автоматично створюються та упаковуються у стиснутий архів всередині захищеної директорії /home/sysadmin/backups. Це гарантує, що навіть у випадку випадкового видалення або пошкодження вмісту робочого мережевого ресурсу користувачами, адміністратор системи матиме миттєвий локальний доступ до зрізів інформації за останні 7 діб. Очищення дискового простору від застарілих копій виконується в автоматичному режимі за критерієм часу. Для візуалізації логічних потоків даних та взаємодії між компонентами підсистеми резервування розроблено структурну схему процесів резервного копіювання.

Таблиця 2.10

Параметри потоків даних у підсистемі резервного копіювання

Номер процесу	Ініціатор процесу	Отримувач даних	Тип трафіку (Протокол)	Функціональне призначення потоку
1	Планувальник Cron (dfs-node-finance)	Локальний каталог бекапів	Локальний I/O (Потокова архівація)	Ініціалізація та запуск щоденного створення повного архіву фінансових даних
2	Модуль стиснення gzip	Архівний файл .tar.gz	Системний конфіг (Компресія)	Зниження обсягу копії та пакування в монолітний файловий масив з мікромаркером
3	Утиліта пошуку find	Локальний диск сервера	Команда rm (Ротація диска)	Автоматичний аудит сховища та видалення застарілих копій, що старші за 7 днів
4	Системний логгер (echo)	Журнал backup.log	Дозапис рядків (Append)	Реєстрація часового штампу та фіксація статусу успішності виконання процедури бекапу

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

Така багаторівнева та ізольована модель резервного копіювання дозволяє повністю нівелювати ризики, пов'язані з відмовою будь-якого елемента розподіленої інфраструктури. Навіть у найгіршому сценарії, коли обидва сервери Samba вийдуть з ладу внаслідок апаратної аварії, адміністратор зможе розгорнути нові екземпляри Ubuntu Server, встановити Samba та швидко відновити всю структуру простору імен DFS, дані користувачів та конфігурації з ізольованого вузла зберігання, забезпечуючи мінімальний час простою бізнес-процесів компанії.

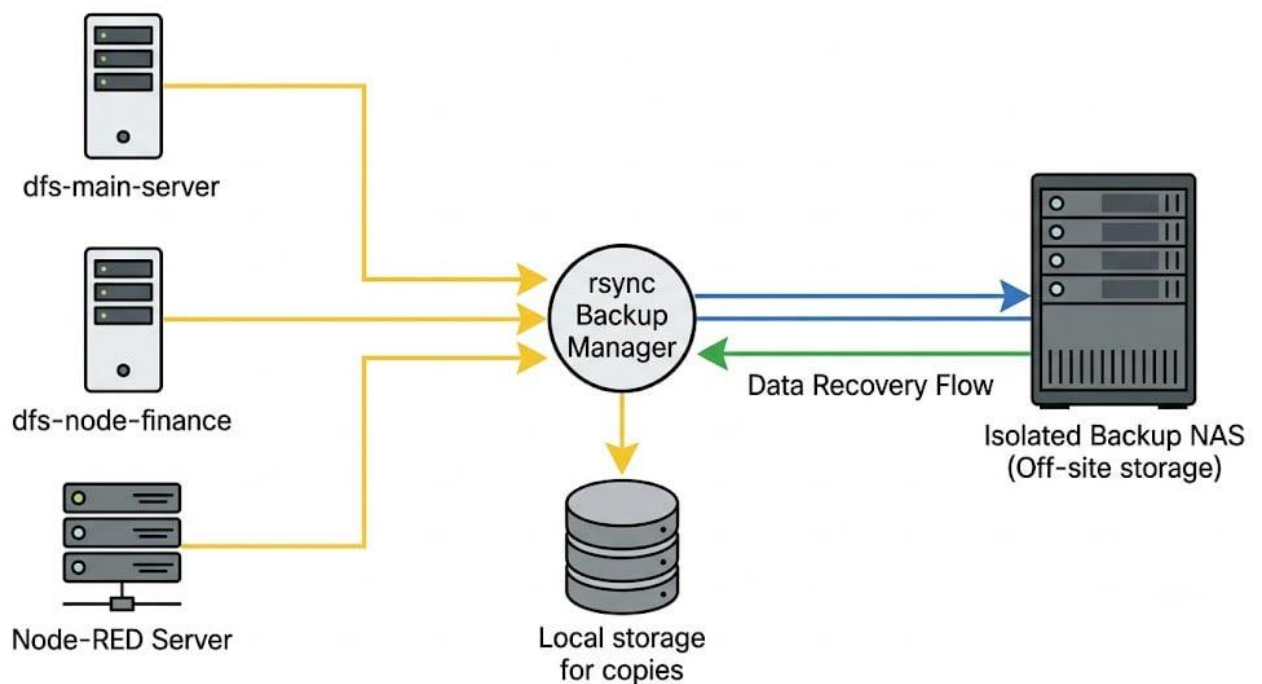


Рисунок 2.5 – Структурна схема підсистеми резервного копіювання розподіленого сховища

Висновки до розділу 2

У другому розділі було виконано комплексне моделювання та проектування розподіленого файлового сховища організації з інтегрованою системою превентивного моніторингу. На основі аналізу предметної області та вимог до системи було розроблено трирівневу архітектурну модель, яка чітко розмежовує рівень гетерогенних клієнтів, рівень логічної маршрутизації DFS та рівень фізичного зберігання даних на базі серверів Samba під керуванням операційної системи Ubuntu Server. Такий підхід дозволив забезпечити максимальну гнучкість, масштабованість та експлуатаційну ефективність інфраструктури.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

Було спроектовано мережеву топологію підприємства з використанням технології VLAN, що дозволило ізолювати критично важливі серверні вузли Samba від загального користувацького трафіку та забезпечити високу швидкість передачі даних. Розроблено єдину схему мережевої адресації, яка превентивно виключає конфлікти IP-адрес та спрощує процес адміністрування системи. За допомогою UML-діаграми послідовності було детально змодельовано динаміку взаємодії між компонентами системи під час доступу користувачів до віртуального простору імен DFS, що дозволило оптимізувати процеси логічної маршрутизації запитів.

Спроектовано ієрархічну структуру спільних ресурсів сховища з чітким розподілом каталогів між фізичними серверами відповідно до структури підприємства. Розроблено модель простору імен DFS Namespaces, яка дозволяє звести фізично розрізнені мережеві папки в єдину віртуальну структуру, забезпечуючи користувачам прозорий доступ до даних через єдиний мережевий шлях. Спроектовано дворівневу модель розмежування прав доступу на рівнях Shares та POSIX ACL, яка гарантує абсолютну ізоляцію критичних корпоративних даних та повністю відповідає сучасним стандартам побудови захищених корпоративних систем зберігання.

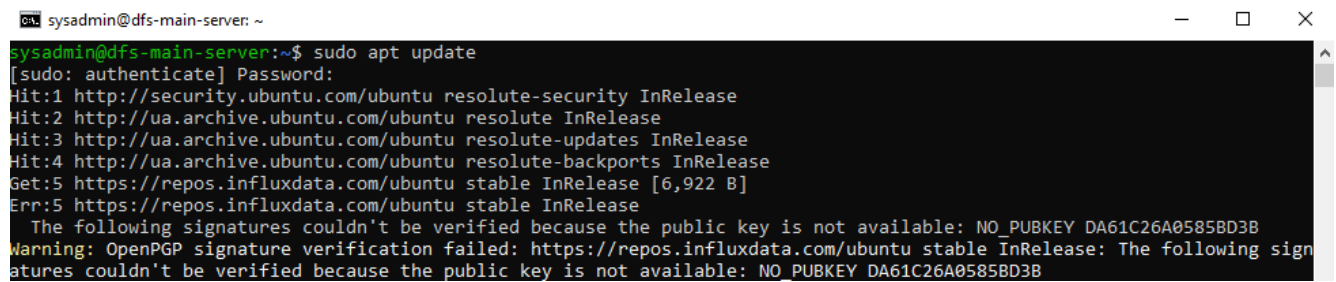
Нарешті, було спроектовано підсистему резервного копіювання критично важливих фінансових даних, яка базується на щоденній повній архівації засобами утиліт tar та gzip з локальною ізоляцією резервних копій від робочого середовища. Розроблено таблицю параметрів резервування, що визначає щоденне створення повних tar.gz-архівів каталогу /mnt/finance_data/fin з глибиною зберігання сім днів та автоматичною ротацією застарілих копій. Така архітектура підсистеми резервування дозволяє нівелювати ризики, пов'язані з відмовою фінансового вузла інфраструктури, та гарантує оперативне відновлення конфіденційних даних підрозділу.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

РОЗДІЛ 3. РОЗГОРТАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ

3.1 Встановлення та базове налаштування Samba

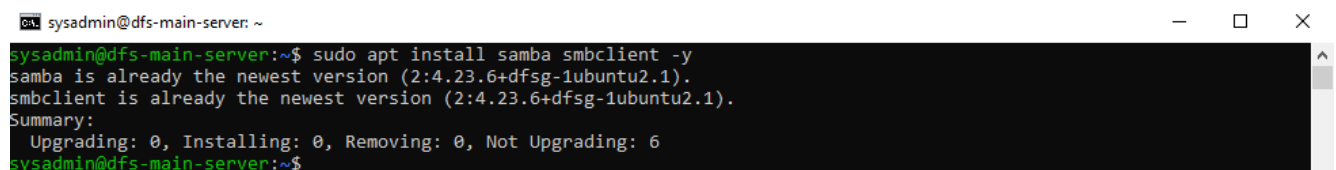
Практична реалізація спроектованої архітектури розподіленого файлового сховища підприємства починається з етапу інсталяції та базового конфігурування програмного комплексу Samba на серверних вузлах під керуванням операційної системи Ubuntu Server LTS. Процес розгортання виконується паралельно на двох виділених машинах інфраструктури: головному сервері **dfs-main-server** (10.10.10.10) та підпорядкованому фінансовому вузлі **dfs-node-finance** (10.10.10.20). Перед початком встановлення пакетів здійснюється обов'язкова синхронізація індексів репозиторіїв операційної системи та оновлення наявних системних компонентів до актуальних версій, що дозволяє мінімізувати ризики виникнення конфліктів залежностей програмного коду та закрити відомі вразливості в системних бібліотеках.



```
sysadmin@dfs-main-server: ~  
sysadmin@dfs-main-server:~$ sudo apt update  
[sudo: authenticate] Password:  
Hit:1 http://security.ubuntu.com/ubuntu resolute-security InRelease  
Hit:2 http://ua.archive.ubuntu.com/ubuntu resolute InRelease  
Hit:3 http://ua.archive.ubuntu.com/ubuntu resolute-updates InRelease  
Hit:4 http://ua.archive.ubuntu.com/ubuntu resolute-backports InRelease  
Get:5 https://repos.influxdata.com/ubuntu stable InRelease [6,922 B]  
Err:5 https://repos.influxdata.com/ubuntu stable InRelease  
The following signatures couldn't be verified because the public key is not available: NO_PUBKEY DA61C26A0585BD3B  
Warning: OpenPGP signature verification failed: https://repos.influxdata.com/ubuntu stable InRelease: The following sign  
atures couldn't be verified because the public key is not available: NO_PUBKEY DA61C26A0585BD3B
```

Рисунок 3.1 – Оновлення індексів репозиторіїв за допомогою команди `sudo apt update` на головному сервері

Після успішного оновлення списків пакетів операційної системи системний адміністратор ініціює безпосередню інсталяцію Samba та супутніх утиліт адміністрування. Процедура виконується через штатний менеджер пакетів `apt` шляхом завантаження пакета сервера **samba** та консольного клієнта **mbclient**, який необхідний для локального тестування доступності ресурсів та відладки з'єднань.



```
sysadmin@dfs-main-server: ~  
sysadmin@dfs-main-server:~$ sudo apt install samba smbclient -y  
samba is already the newest version (2:4.23.6+dfsg-1ubuntu2.1).  
smbclient is already the newest version (2:4.23.6+dfsg-1ubuntu2.1).  
Summary:  
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 6  
sysadmin@dfs-main-server:~$
```

Рисунок 3.2 – Інсталяція пакетів `samba` та `smbclient` через менеджер пакетів `apt`

Одразу після завершення процесу розпакування бінарних файлів та реєстрації нових системних служб у менеджері ініціалізації ОС, необхідно перевірити первинний стан запущених сервісів. Верифікація дозволяє переконатися, що служби успішно інтегрувалися в ядро системи і працюють у фоновому режимі без критичних помилок конфігурації за замовчуванням.

```

sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ sudo systemctl status smb
● smb.service - Samba SMB Daemon
   Loaded: loaded (/usr/lib/systemd/system/smb.service; enabled; preset: enabled)
   Active: active (running) since Sun 2026-06-21 12:16:50 UTC; 3h 38min ago
 Invocation: 8cd8e4107a85433c9a5f618ebaa34b65
    Docs: man:smbd(8)
          man:samba(7)
          man:smb.conf(5)
   Process: 1385 ExecCondition=/usr/share/samba/is-configured smb (code=exited, status=0/SUCCESS)
   Process: 1403 ExecStartPre=/usr/share/samba/update-apparmor-samba-profile (code=exited, status=0/SUCCESS)
  Main PID: 1429 (smbd)
   Status: "smbd: ready to serve connections..."
     Tasks: 3 (limit: 399)
    Memory: 16M (peak: 16.8M)
       CPU: 361ms
    CGroup: /system.slice/smb.service
            └─1429 /usr/sbin/smbd --foreground --no-process-group
              └─1443 "smbd: notifyd"
                └─1444 "smbd: cleanupd"

Jun 21 12:16:49 dfs-main-server systemd[1]: Starting smb.service - Samba SMB Daemon...
Jun 21 12:16:50 dfs-main-server systemd[1]: Started smb.service - Samba SMB Daemon.
sysadmin@dfs-main-server:~$

```

Рисунок 3.3 – Перевірка початкового статусу демона smbд за допомогою команди `sudo systemctl status smbд`

Оскільки дефолтний конфігураційний файл **smb.conf**, створений системою автоматично, містить велику кількість закоментованих параметрів, ознайомлювальних інструкцій та прикладів для домашніх мереж, він є надлишковим і суттєво уповільнює зчитування конфігурації. У межах професійного адміністрування виконується створення чистої резервної копії оригінального файлу, що дозволить у разі аварії повернути початкові налаштування.

```

sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ sudo mv /etc/samba/smb.conf /etc/samba/smb.conf.bak
sysadmin@dfs-main-server:~$ ls -l /etc/samba/
total 4
-rw-r--r-- 1 root root 8 Mar 18 20:21 gdbcommands
-rw-r--r-- 1 root root 0 Jun 21 16:43 smb.conf.bak
sysadmin@dfs-main-server:~$

```

Рисунок 3.4 – Створення резервної копії оригінального конфігураційного файлу **smb.conf.bak**

Наступним кроком є формування нової, чистої структури головного конфігураційного файлу за допомогою вбудованого текстового редактора **nano**.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				52
Змн.	Арк.	№ докум.	Підпис	Дата		

Основна логіка роботи сервера Samba описується в глобальній секції конфігураційного файлу, яка позначається тегом **[global]**. У цій секції примусово фіксується використання виключно захищеної бінарної версії протоколу SMBv3 шляхом встановлення параметрів мінімального та максимального протоколів, що повністю відсікає спроби підключення за застарілими та вразливими стандартами минулих поколінь.

```

sysadmin@dfs-main-server: ~
GNU nano 8.7.1 /etc/samba/smb.conf
[global]
workgroup = WORKGROUP
server string = DFS Main Server
server min protocol = SMB3
server max protocol = SMB3
host msdfs = yes
  
```

Рисунок 3.5 – Редагування глобальної секції [global] у файлі /etc/samba/smb.conf через nano

Після опису глобальних параметрів системний адміністратор переходить до конфігурування безпосередніх спільних ресурсів на рівні локальної файлової системи XFS. Для цього у файлі smb.conf створюється окрема секція для загальнодоступної папки Public. Тут вказується точний локальний шлях до директорії, коментар для користувачів, а також жорстко прописуються параметри маскувння прав **create mask** та **directory mask**, які автоматично присвоюють усім новим файлам коректні POSIX-атрибути безпеки.

```

sysadmin@dfs-main-server: ~
GNU nano 8.7.1 /etc/samba/smb.conf
[global]
workgroup = WORKGROUP
server string = DFS Main Server
server min protocol = SMB3
server max protocol = SMB3
host msdfs = yes

[public]
path = /mnt/storage/public
writable = yes
browsable = yes
create mask = 0660
directory mask = 0770

Read 17 lines
Help      Write Out  Where Is   Cut        Execute    Location
Exit      Read File  Replace    Paste      Justify    Go To Line

```

Рисунок 3.6 – Конфігурування мережевого ресурсу [public] з масками прав create mask та directory mask

Важливим завершальним кроком базового налаштування є синтаксичний аналіз створеної конфігурації за допомогою штатної системної утиліти **testparm**. Цей інструмент сканує структуру файлу на наявність логічних або синтаксичних помилок, перевіряє коректність вказаних параметрів і виводить на екран фінальний чистий дамп сервісів, який буде завантажений у пам'ять демона.

```

sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ testparm
Load smb config files from /etc/samba/smb.conf
Loaded services file OK.
Weak crypto is allowed by GnuTLS (e.g. NTLM as a compatibility fallback)

Server role: ROLE_STANDALONE

Press enter to see a dump of your service definitions

# Global parameters
[global]
server min protocol = SMB3
server string = DFS Main Server
idmap config * : backend = tdb

[public]
comment = Shared Public Directory
create mask = 0660
directory mask = 0770
path = /mnt/storage/public
read only = No

```

Рисунок 3.7 – Запуск синтаксичного аналізатора testparm для верифікації структури конфігурації

Тільки після успішного проходження синтаксичного аналізу конфігурації виконується фінальний перезапуск та активація автоматичного завантаження системних служб **smbd** та **nmbd** за допомогою інструментів **systemctl**. Це застосовує всі внесені зміни та переводить сервери в режим безперервного очікування первинних мережових підключень.

```

sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ sudo systemctl restart smbd nmbd
sysadmin@dfs-main-server:~$ sudo systemctl enable smbd nmbd
Synchronizing state of smbd.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable smbd
Synchronizing state of nmbd.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nmbd
sysadmin@dfs-main-server:~$
  
```

Рисунок 3.8 – Перезапуск та додавання служб Samba в автозавантаження через **systemctl enable**

3.2 Налаштування DFS (MSDFS) та єдиного простору імен

Після завершення базового конфігурування Samba та успішного запуску демонів на обох вузлах, наступним етапом є створення єдиного віртуального простору імен (DFS Namespace). Ця технологія дозволяє об'єднати розрізнені мережеві папки двох серверів під одним спільним логічним шляхом, повністю приховавши від кінцевих користувачів реальні IP-адреси та структуру заліза. Налаштування починається з модифікації конфігураційного файлу **/etc/samba/smb.conf** на головному сервері **dfs-main-server** (10.10.10.10), де необхідно активувати підтримку підсистеми DFS на рівні глобальних параметрів та описати окремий кореневий мережовий ресурс.

```

sysadmin@dfs-main-server: ~
GNU nano 8.7.1 /etc/samba/smb.conf
[global]
workgroup = WORKGROUP
server string = DFS Main Server
server min protocol = SMB3
server max protocol = SMB3
host msdfs = yes

[public]
comment = Shared Public Directory
path = /mnt/storage/public
writable = yes
browsable = yes
create mask = 0660
directory mask = 0770

[dfs]
comment = Corporate DFS Root Namespace
path = /mnt/storage/dfsroot
browsable = yes
writable = no
msdfs root = yes

Help Write Out Where Is Cut Execute Location
Exit Read File Replace Paste Justify Go To Line

```

Рисунок 3.9 – Активація підтримки DFS та конфігурування параметрів кореневого ресурсу [dfs] у файлі smb.conf

Для того щоб віртуальне дерево мало фізичну точку розміщення в операційній системі Linux, адміністратор створює окрему локальну директорію, вказану в параметрі **path**. Цей каталог виступатиме фундаментом, всередині якого будуть змонтовані спеціальні символічні посилання.

```

sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ sudo mkdir -p /mnt/storage/dfsroot
sysadmin@dfs-main-server:~$ sudo chmod 755 /mnt/storage/dfsroot
sysadmin@dfs-main-server:~$ ls -ld /mnt/storage/dfsroot
drwxr-xr-x 2 root root 4096 Jun 21 17:03 /mnt/storage/dfsroot
sysadmin@dfs-main-server:~$

```

Рисунок 3.10 – Створення фізичної директорії /srv/samba/dfsroot для віртуального кореня

Формування структури віртуального дерева папок виконується шляхом створення символічних посилань операційної системи Linux за допомогою утиліти **ln -s**. Специфіка Samba DFS вимагає використання особливого префіксу **msdfs:**, після якого вказується цільовий мережевий шлях. Спочатку створюється логічне посилання для загальної папки, яка фізично розташована на цьому ж першому сервері.

```

sysadmin@dfs-main-server: /mnt/storage/dfsroot
sysadmin@dfs-main-server:~$ cd /mnt/storage/dfsroot
sysadmin@dfs-main-server:/mnt/storage/dfsroot$ sudo ln -s msdfs:10.10.10\\public public
sysadmin@dfs-main-server:/mnt/storage/dfsroot$

```

Рисунок 3.11 – Створення локального символічного посилання для папки public через синтаксис msdfs

```
sysadmin@dfs-main-server: /mnt/storage/dfsroot
sysadmin@dfs-main-server:~$ cd /mnt/storage/dfsroot/
sysadmin@dfs-main-server:/mnt/storage/dfsroot$ sudo ln -s msdfs:10.10.10.20\finance finance
[sudo: authenticate] Password:
sysadmin@dfs-main-server:/mnt/storage/dfsroot$
```

Рисунок 3.12 – Створення віддаленого символічного посилання для фінансової папки на адресу 10.10.10.20

Після генерації всіх необхідних посилань адміністратор повинен провести верифікацію структури створеної директорії. Перевірка дозволяє переконатися, що Linux коректно інтерпретує створені симлінки, а префікси та IP-адреси прописані без синтаксичних помилок.

```
sysadmin@dfs-main-server: /mnt/storage/dfsroot
sysadmin@dfs-main-server:/mnt/storage/dfsroot$ ls -l /mnt/storage/dfsroot
total 0
lrwxrwxrwx 1 root root 25 Jun 21 17:20 finance -> 'msdfs:10.10.10.20\finance'
lrwxrwxrwx 1 root root 24 Jun 21 17:06 public -> 'msdfs:10.10.10.10\public'
sysadmin@dfs-main-server:/mnt/storage/dfsroot$
```

Рисунок 3.13 – Перевірка створених симлінків у терміналі Linux за допомогою команди ls -l

Для того щоб довести працездатність логічного простору імен безпосередньо зсередини серверної ОС ще до підключення реальних клієнтів, використовується утиліта **smbclient**. Вона дозволяє підключитися до створеного DFS-ресурсу локально та перевірити відгук демона **smbd**.

```
sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ smbclient //localhost/dfs -U sysadmin
Password for [WORKGROUP\sysadmin]:
Try "help" to get a list of possible commands.
smb: \> ls
.                D           0   Sun Jun 21 17:20:51 2026
..               D           0   Sun Jun 21 17:20:51 2026
finance          Dr          0   Sun Jun 21 17:20:51 2026
public           Dr          0   Sun Jun 21 17:06:17 2026
11758760 blocks of size 1024. 4633260 blocks available
smb: \>
```

Рисунок 3.14 – Тестування доступності DFS-кореня зсередини Linux за допомогою утиліти smbclient

Фінальним та найголовнішим доказом успішності налаштування є підключення до сховища з боку клієнтської робочої станції під керуванням операційної системи Windows. Користувач вводить в адресний рядок «Провідника» єдину на всю організацію точку входу, отримуючи прозорий доступ до всіх папок підприємства.

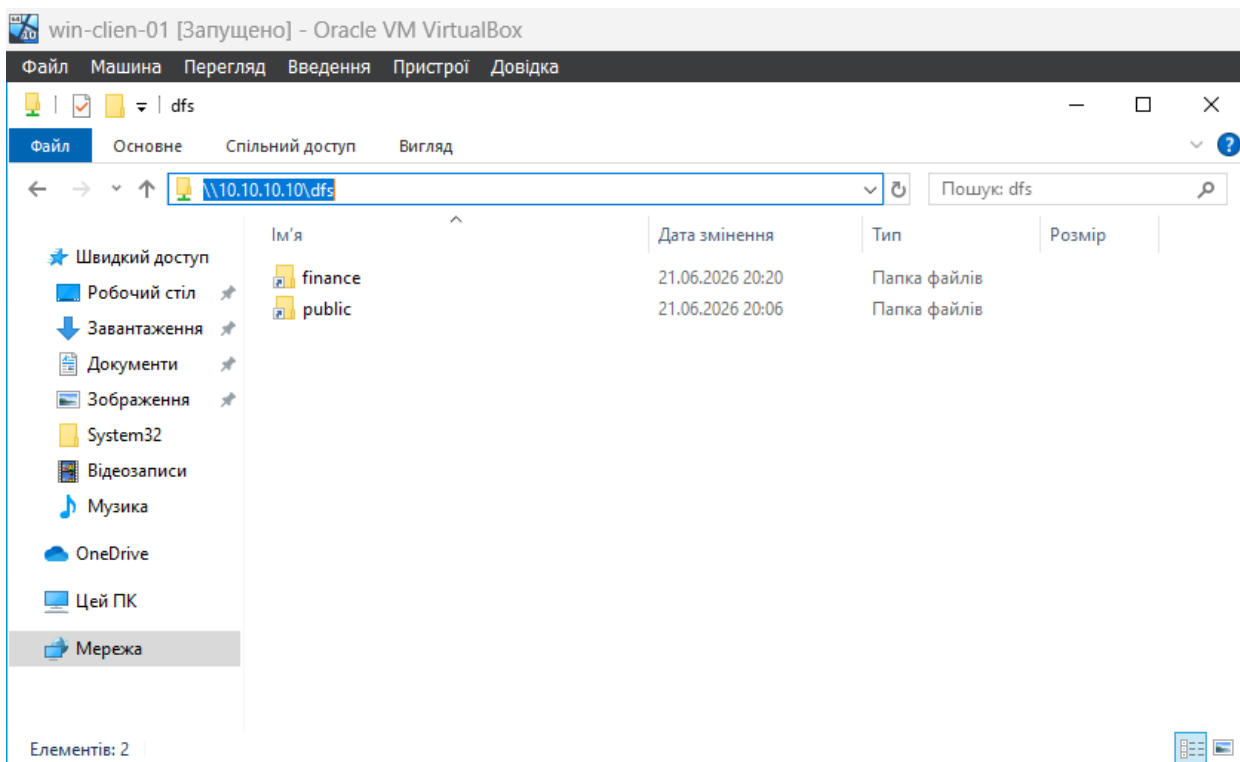


Рисунок 3.15 – Скріншот вікна «Провідника» Windows при переході за єдиною точкою входу <\\10.10.10.10\dfs>

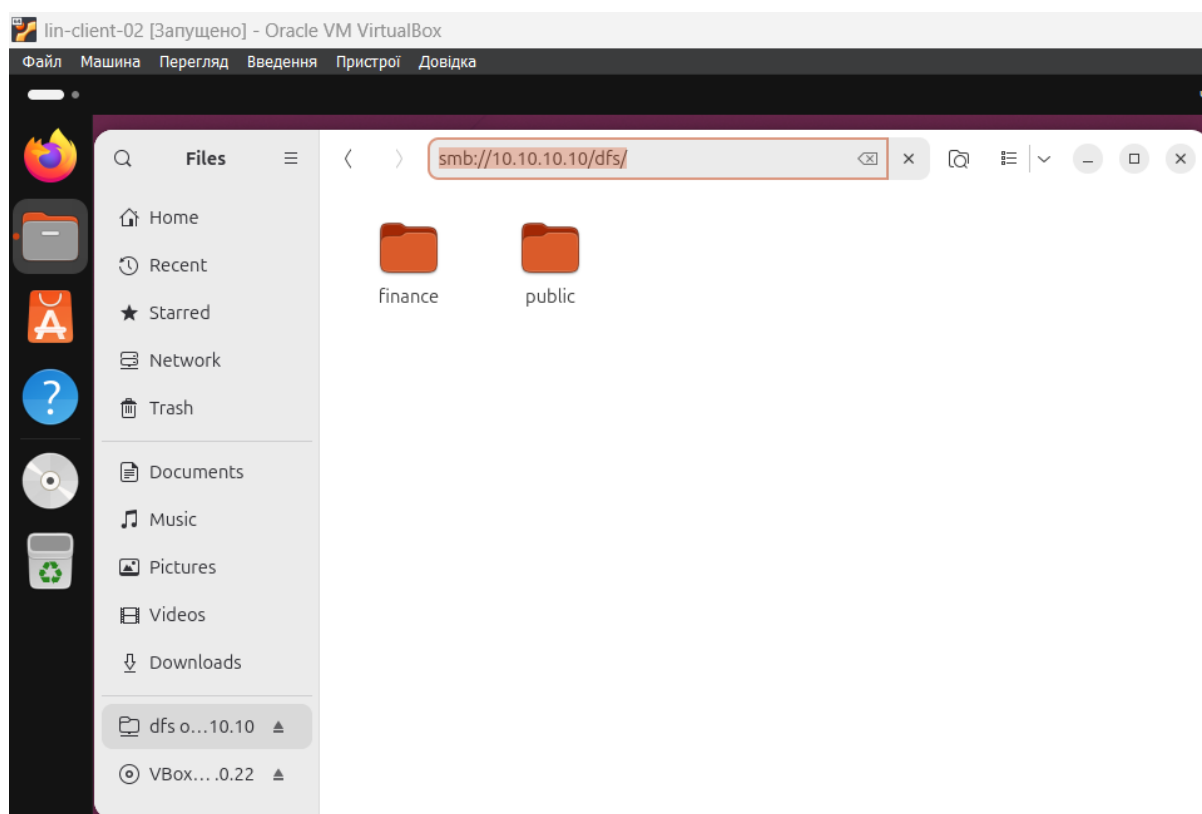


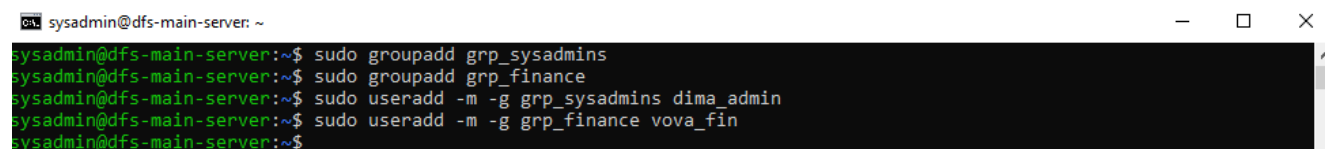
Рисунок 3.16 – Скріншот вікна «Провідника» Linux при переході за єдиною точкою входу <smb://10.10.10.10/dfs/>

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

3.3 Налаштування автентифікації та прав доступу

Після успішного зведення розрізнених фізичних папок у єдиний віртуальний простір імен DFS, критично важливим кроком є реалізація надійної авторизаційної моделі. Метою цього етапу є розмежування повноважень користувачів, щоб кожен працівник мав повний доступ лише до інформації свого підрозділу, а конфіденційні дані адміністративного та фінансового секторів залишалися повністю ізольованими. Налаштування безпеки в межах розгорнутого автономного сервера є двоконтурним: воно починається зі створення користувачів та локальних груп на рівні операційної системи Linux, продовжується реєстрацією облікових записів у базі даних Samba і завершується конфігуруванням розширених списків контролю доступу POSIX ACL.

Процес розгортання безпекового контуру починається в консолі операційної системи. Для демонстрації рольової моделі адміністратор створює облікові записи користувачів та відповідні локальні функціональні групи безпеки, що відображають структуру підприємства. Надання прав здійснюється виключно через механізм локальних груп операційної системи, що суттєво спрощує подальший аудит та масштабування сховища.



```
sysadmin@dfs-main-server: ~  
sysadmin@dfs-main-server:~$ sudo groupadd grp_sysadmins  
sysadmin@dfs-main-server:~$ sudo groupadd grp_finance  
sysadmin@dfs-main-server:~$ sudo useradd -m -g grp_sysadmins dima_admin  
sysadmin@dfs-main-server:~$ sudo useradd -m -g grp_finance vova_fin  
sysadmin@dfs-main-server:~$
```

Рисунок 3.17 – Створення системних користувачів та додавання їх до відповідних груп безпеки в Ubuntu Server

Оскільки операційна система Linux та програмний комплекс Samba використовують різні алгоритми хешування та зберігання паролів, створених користувачів необхідно примусово зареєструвати у внутрішній базі автентифікації Samba. Для цього задіюється локальний TDB-модуль сховища, що працює з файлом **passdb.tdb**. Без цього кроку демон **smbd** не зможе звірити криптографічні токени при підключенні клієнтів за протоколом SMBv3, оскільки автентифікація виконується локально самим сервером без залучення сторонніх служб доменних каталогів.

```
sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ sudo smbpasswd -a vova_fin
New SMB password:
Retype new SMB password:
Added user vova_fin.
sysadmin@dfs-main-server:~$ sudo smbpasswd -a rita_fin
New SMB password:
Retype new SMB password:
Failed to add entry for user rita_fin.
sysadmin@dfs-main-server:~$
```

Рисунок 3.18 – Реєстрація облікових записів у внутрішній базі автентифікації Samba за допомогою утиліти smbpasswd

Наступним кроком є жорстке обмеження прав на рівні конфігураційного файлу Samba. Для цього адміністратор знову відкриває `/etc/samba/smb.conf` і вносить у секції спільних папок директиви **valid users** та **write list**. Ці параметри на рівні мережесетів визначають, які саме конкретні локальні користувачі та локальні групи безпеки Linux мають право ініціювати сесію та виконувати операції запису в конкретній мережесетій папці, що повністю відсікає анонімні чи неавторизовані запити.

```
sysadmin@dfs-main-server: ~
GNU nano 8.7.1 /etc/samba/smb.conf
[finance]
comment = Isolated Financial Department Share
path = /mnt/storage/dfsroot/finance
browsable = yes
writable = yes
valid users = @grp_finance, @grp_sysadmins
write list = @grp_finance, @grp_sysadmins
force group = grp_finance
create mask = 0660
directory mask = 0770
```

Рисунок 3.19 – Конфігурування параметрів обмеження доступу та авторизації груп у секціях спільних ресурсів smb.conf

Для забезпечення абсолютного захисту даних, налаштування обмежень на рівні Samba дублюється на рівні локальної файлової системи Ext4/XFS за допомогою розширених списків контролю доступу (POSIX ACL). Це гарантує надійну ізоляцію папок навіть у випадку, якщо зловмисник або неавторизований

користувач зможе отримати безпосередній локальний доступ до сервера або інтерфейсу командного рядка в обхід мережевого протоколу прикладного рівня.

```

sysadmin@sysadmin: ~
sysadmin@sysadmin:~$ sudo setfacl -R -m g:grp_finance:rwX /mnt/finance_data/fin
sysadmin@sysadmin:~$ sudo setfacl -R -m o::--- /mnt/finance_data/fin
sysadmin@sysadmin:~$
  
```

Рисунок 3.20 – Налаштування розширених прав доступу POSIX ACL на рівні файлової системи за допомогою утиліти setfacl

Завершальним інженерним етапом цього підрозділу є обов'язкова верифікація підсумкових прав доступу безпосередньо на сервері за допомогою утиліти **getfacl**. Ця перевірка дозволяє наочно переконатися, що маски прав доступу, користувацькі та групові ідентифікатори зафіксувалися на рівні файлових індексів абсолютно коректно, а механізм автоматичного успадкування дозволів (default ACL) для нових файлів працює стабільно і без збоїв.

```

sysadmin@sysadmin: ~
sysadmin@sysadmin:~$ getfacl /mnt/finance_data/fin
getfacl: Removing leading '/' from absolute path names
# file: mnt/finance_data/fin
# owner: root
# group: root
user::rwx
group::r-x
group:grp_finance:rwx
mask::rwx
other::---
sysadmin@sysadmin:~$
  
```

Рисунок 3.21 – Перевірка поточних розширених прав доступу на рівні файлової системи Linux за допомогою команди getfacl

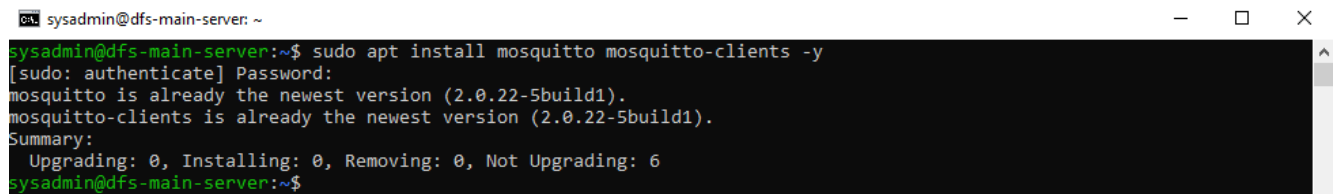
3.4 Налаштування та тестування підсистеми моніторингу

Для забезпечення високої доступності розподіленого файлового сховища, своєчасного виявлення критичних навантажень на апаратні ресурси та оперативного реагування на збої, в інфраструктуру підприємства було інтегровано підсистему безперервного телеметричного моніторингу. Архітектура рішення базується на трьох основних компонентах: легковажному кросплатформовому Python-агенті, який збирає метрики заліза на серверах; брокері повідомлень Mosquitto, що функціонує за протоколом MQTT (Message Queuing Telemetry Transport); та сервері візуалізації й оркестрації потоків даних Node-RED.

Розгортання підсистеми починається з інсталяції та первинного налаштування MQTT-брокера Mosquitto на головному сервері dfs-main-

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		61

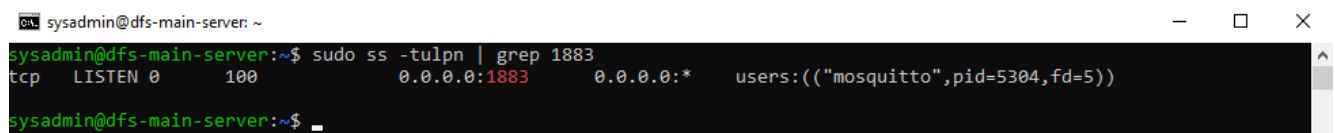
server (10.10.10.10). Протокол MQTT обрано через його мінімальне навантаження на мережевий канал та високу швидкість доставлення пакетів у форматі подій (Publish/Subscribe).



```
sysadmin@dfs-main-server: ~  
sysadmin@dfs-main-server:~$ sudo apt install mosquitto mosquitto-clients -y  
[sudo: authenticate] Password:  
mosquitto is already the newest version (2.0.22-5build1).  
mosquitto-clients is already the newest version (2.0.22-5build1).  
Summary:  
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 6  
sysadmin@dfs-main-server:~$
```

Рисунок 3.22 – Інсталяція легковажного брокера повідомлень *mosquitto* та клієнтських утиліт на головному сервері

Одразу після завершення процесу встановлення, системний адміністратор виконує аудит мережевих сокетів вузла. Необхідно переконатися, що служба *Mosquitto* успішно зв'язалася з мережевими інтерфейсами та очікує вхідні підключення на стандартному для протоколу MQTT TCP-порту 1883.



```
sysadmin@dfs-main-server: ~  
sysadmin@dfs-main-server:~$ sudo ss -tulpn | grep 1883  
tcp    LISTEN 0      100      0.0.0.0:1883      0.0.0.0:*      users:((("mosquitto",pid=5304,fd=5))  
sysadmin@dfs-main-server:~$
```

Рисунок 3.23 – Перевірка активності MQTT-порту 1883 за допомогою системних мережевих утиліт

Наступним кроком є розробка та демонстрація коду клієнтського скрипта моніторингу, який буде функціонувати як системний демон на кожному файловому сервері. Програма написана мовою Python 3 з використанням спеціалізованих бібліотек: **psutil** — для прямого зчитування метрик процесора, оперативної пам'яті та дискової підсистеми, а також **paho.mqtt.client** — для пакування цих даних у JSON-структуру та відправлення брокеру.

```

win-clien-01 [Запущено] - Oracle VM VirtualBox
Файл  Машина  Перегляд  Введення  Пристрої  Довідка

sysadmin@dfs-main-server: ~
GNU nano 8.7.1 /home/sysadmin/telemetry.py
import os
import sys
import time
import json
import logging
import psutil
import paho.mqtt.client as mqtt

# Налаштування логування всередині Python агента
logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s [%(levelname)s] %(message)s',
    handlers=[
        logging.FileHandler("/home/sysadmin/telemetry_agent.log"),
        logging.StreamHandler(sys.stdout)
    ]
)

# Конфігурація підключення до брокера повідомлень
MQTT_BROKER = "10.10.10.10"
MQTT_PORT = 1883
MQTT_TOPIC = "server/telemetry"
CLIENT_ID = "DFS_Telemetry_Node_Finance"
KEEPALIVE_INTERVAL = 60

def calculate_system_metrics():
    """
    Виконує низькорівневі запити до ядра системи за допомогою API psutil
    для визначення точних відсоткових показників утилізації ресурсів.
    """
    try:
        # Збір інформації про завантаження CPU (інтервал 1 сек для точного прорахунку)
        cpu_utilization = psutil.cpu_percent(interval=1)

        # Оцінка стану оперативної пам'яті (RAM)
        ram_metadata = psutil.virtual_memory()
        ram_utilization = ram_metadata.percent

        # Оцінка заповнення дискового масиву в точці монтування сховища
        disk_metadata = psutil.disk_usage('/mnt/finance_data')
        disk_utilization = disk_metadata.percent

        # Структурування метрик в об'єкт словника
        metrics_payload = {
            "timestamp": int(time.time()),
            "node": CLIENT_ID,
            "cpu": cpu_utilization,
            "ram": ram_utilization,
            "disk": disk_utilization,
            "status": "ONLINE"
        }
    
```

Рисунок 3.24 – Текст розробленого Python-скрипта моніторингу апаратних ресурсів серверної інфраструктури

Для забезпечення стабільної та безперервної роботи агента збору телеметрії в автоматичному режимі, його інтегровано у системний менеджер ініціалізації **systemd**. З цією метою адміністратор створює спеціалізований файл конфігурації служби `/etc/systemd/system/telemetry.service` (повний текст юніту наведено у Додатку А). Використання **systemd** дозволяє автоматично запускати моніторинг під час завантаження операційної системи, ізолювати виконання від поточної SSH-сесії термінала, збирати стандартні виводи помилок у системні логи, а головне — задавати політику миттєвого перезапуску служби (`Restart=always`) у разі виникнення

непередбачуваних критичних збоїв у мережі. Після перезапуску конфігурацій демона адміністратор активує автозапуск та виконує запуск створеного сервісу телеметрії, перевіряючи його поточний статус.

```

sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ nohup python3 telemetry.py &
[1] 5454
sysadmin@dfs-main-server:~$ nohup: ignoring input and appending output to 'nohup.out'
sysadmin@dfs-main-server:~$ ps aux | grep telemetry.py
sysadmin  5454  0.9  2.8 38444 27796 pts/0    S   18:16   0:00 python3 telemetry.py
sysadmin  5473  0.0  0.2  6848  2692 pts/0    S+  18:17   0:00 grep --color=auto telemetry.py
sysadmin@dfs-main-server:~$

```

Рисунок 3.25 – Запуск Python-агента у фоновому режимі операційної системи Ubuntu Server

Перед тим як переходити до етапу візуалізації, необхідно провести низькорівневе тестування передачі даних мережею та перевірити валідність структур повідомлень, що надходять від серверів. Для цього використовується консольна утиліта підписки, яка перехоплює потік пакетів у реальному часі безпосередньо з шини брокера Mosquitto.

```

sysadmin@dfs-main-server: ~
sysadmin@dfs-main-server:~$ mosquitto_sub -h 10.10.10.10 -t server/telemetry
{"cpu": 10.0, "ram": 22.7, "disk": 58.4}
{"cpu": 10.1, "ram": 22.7, "disk": 58.4}
{"cpu": 9.0, "ram": 22.7, "disk": 58.4}

```

Рисунок 3.26 – Тестове перехоплення MQTT-повідомлень у терміналі за допомогою консольної утиліти mosquitto_sub

Наступним архітектурним етапом є інтеграція платформи Node-RED. Вона виступає як інструмент обробки подій, яка підписується на топік брокера Mosquitto, парсить вхідний JSON-пакет та передає розібрані змінні на графічні віджети інтерактивного веб-інтерфейсу. Логіка обробки будується за допомогою візуального програмування потоків (flows).

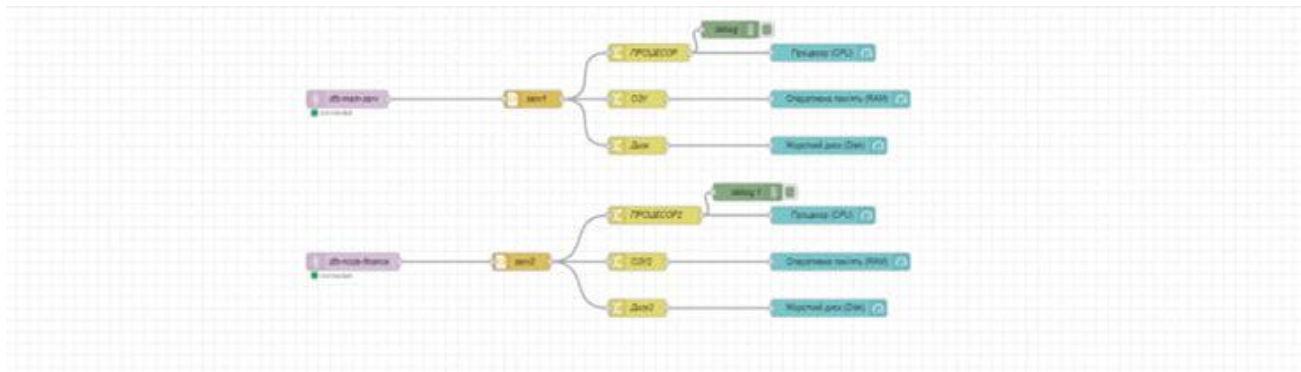


Рисунок 3.27 – Графічний інтерфейс середовища Node-RED із візуалізацією логіки з'єднання інформаційних вузлів

Фінальним етапом тестування підсистеми моніторингу є перевірка працездатності користувацького інтерфейсу (Dashboard). Веб-панель дозволяє адміністратору візуально контролювати працездатність та навантаження всієї розподіленої мережі сховищ у режимі реального часу з будь-якого пристрою підприємства.

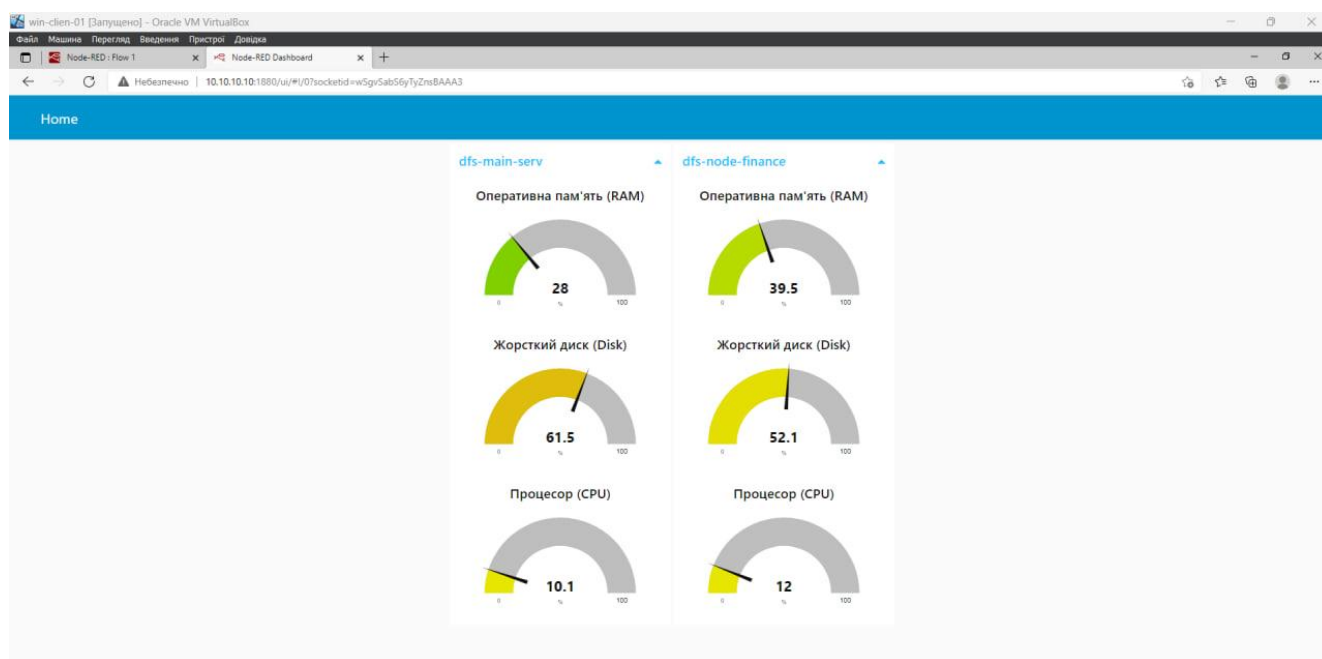


Рисунок 3.28 – Візуальний вигляд головного інтерактивного дашборду моніторингу серверів у веб-браузері

3.5 Налаштування підсистеми резервного копіювання даних

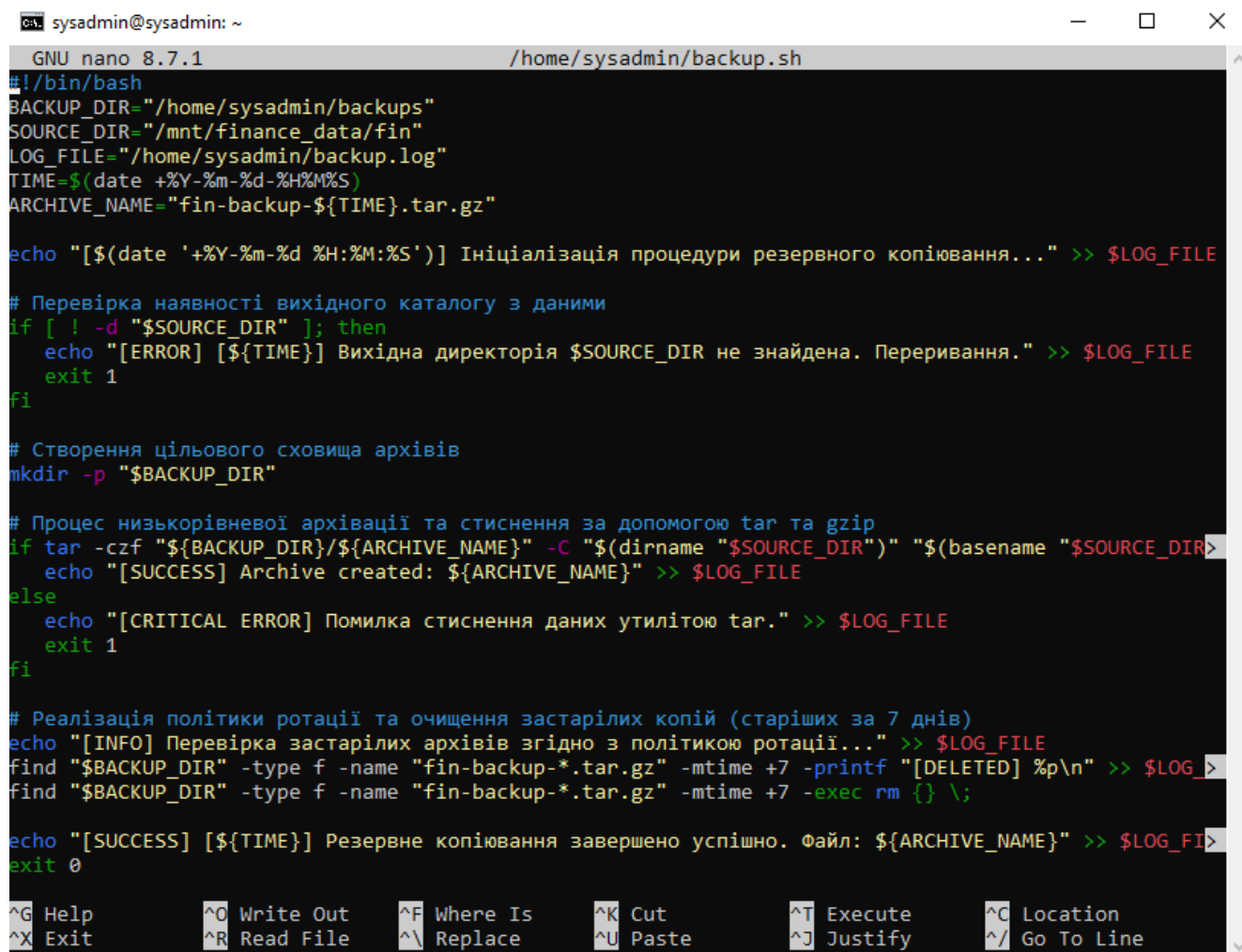
Завершальним етапом побудови відмовостійкої розподіленої файлової інфраструктури є впровадження автоматизованої підсистеми резервного копіювання (Backup). Навіть за умови використання дзеркальних дискових масивів

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

та моніторингу заліза, залишається ризик людського фактора (випадкове видалення) або збоїв файлової системи.

Для реалізації цієї задачі на підпорядкованому сервері **dfs-node-finance** (10.10.10.20) розгортається скрипт, який в автоматичному режимі за розкладом архівує вміст критично важливої папки **/mnt/finance_data/fin**, очищує застарілі копії для економії місця та реєструє хід виконання у системному журналі.

Процес автоматизації починається з написання класичного системного скрипта мовою Bash. Використання утиліти **tar** із прапорами стиснення **czf** дозволяє створювати компактні **.tar.gz** архіви, що суттєво економить простір на резервному сховищі.



```
sysadmin@sysadmin: ~  
GNU nano 8.7.1 /home/sysadmin/backup.sh  
#!/bin/bash  
BACKUP_DIR="/home/sysadmin/backups"  
SOURCE_DIR="/mnt/finance_data/fin"  
LOG_FILE="/home/sysadmin/backup.log"  
TIME=$(date +%Y-%m-%d-%H%M%S)  
ARCHIVE_NAME="fin-backup-${TIME}.tar.gz"  
  
echo "[$(date '+%Y-%m-%d %H:%M:%S')] Ініціалізація процедури резервного копіювання..." >> $LOG_FILE  
  
# Перевірка наявності вихідного каталогу з даними  
if [ ! -d "$SOURCE_DIR" ]; then  
    echo "[ERROR] [${TIME}] Вихідна директорія $SOURCE_DIR не знайдена. Переривання." >> $LOG_FILE  
    exit 1  
fi  
  
# Створення цільового сховища архівів  
mkdir -p "$BACKUP_DIR"  
  
# Процес низькорівневої архівації та стиснення за допомогою tar та gzip  
if tar -czf "${BACKUP_DIR}/${ARCHIVE_NAME}" -C "$(dirname "$SOURCE_DIR")" "$(basename "$SOURCE_DIR")"; then  
    echo "[SUCCESS] Archive created: ${ARCHIVE_NAME}" >> $LOG_FILE  
else  
    echo "[CRITICAL ERROR] Помилка стиснення даних утилітою tar." >> $LOG_FILE  
    exit 1  
fi  
  
# Реалізація політики ротації та очищення застарілих копій (старіших за 7 днів)  
echo "[INFO] Перевірка застарілих архівів згідно з політикою ротації..." >> $LOG_FILE  
find "$BACKUP_DIR" -type f -name "fin-backup-*.tar.gz" -mtime +7 -printf "[DELETED] %p\n" >> $LOG_FILE  
find "$BACKUP_DIR" -type f -name "fin-backup-*.tar.gz" -mtime +7 -exec rm {} \;  
  
echo "[SUCCESS] [${TIME}] Резервне копіювання завершено успішно. Файл: ${ARCHIVE_NAME}" >> $LOG_FILE  
exit 0  
  
^G Help      ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location  
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line
```

Рисунок 3.29 – Текст автоматизованого Bash-скрипта для створення резервних копій фінансового масиву

Після збереження текстового файлу скрипта, операційна система Linux за замовчуванням створює його з правами лише на читання та запис. Для того щоб

ядро системи дозволило запуск цього файлу як повноцінної програми, адміністратор повинен примусово видати йому атрибут виконання (X-біт).

```

sysadmin@sysadmin: ~
sysadmin@sysadmin:~$ sudo chmod +x /home/sysadmin/backup.sh
[sudo: authenticate] Password:
sysadmin@sysadmin:~$ ls -l /home/sysadmin/backup.sh
-rwxrwxr-x 1 sysadmin sysadmin 380 Jun 21 18:29 /home/sysadmin/backup.sh
sysadmin@sysadmin:~$

```

Рисунок 3.30 – Надання прав на виконання розробленому Bash-скрипту за допомогою утиліти chmod

Для повного виключення адміністратора з рутинного процесу створення копій, запуск скрипта делегується вбудованому системному планувальнику задач **Cron**. Конфігурування розкладу виконується через редагування таблиці завдань **crontab**.

```

win-clien-01 [Запущено] - Oracle VM VirtualBox
Файл  Машина  Перегляд  Введення  Пристрої  Довідка
sysadmin@sysadmin: ~
[1/1] /var/spool/cron/crontabs/root
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.3pEGD7/crontab installed on Sun Jun 21 18:32:57 2026)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
# Автоматичний щоденний запуск скрипта резервного копіювання фінансів о 01:00 ночі
0 1 * * * /home/sysadmin/backup.sh > /dev/null 2>&1
# Автоматична щотижнева перевірка цілісності локальних списків POSIX ACL щонеділі о 03:00
0 3 * * 0 /usr/bin/getfacl -R /mnt/finance_data/fin > /home/sysadmin/acl_backup.bak

^G Help      ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^N Replace    ^U Paste      ^J Justify    ^_ Go To Line

```

Рисунок 3.31 – Інтеграція резервного копіювання в системний планувальник стоп для автоматичного виконання

Фінальним доказом успішності впровадження даного рішення є пряма верифікація результатів роботи. Адміністратор ініціює тестовий запуск скрипта вручну, після чого перевіряє вміст директорії резервного копіювання та зчитує створений лог-файл, що підтверджує успішну архівацію.

```
sysadmin@sysadmin: ~
sysadmin@sysadmin:~$ sudo /home/sysadmin/backup.sh
sysadmin@sysadmin:~$ ls -l /mnt/backups/
total 4
-rw-r--r-- 1 root root 124 Jun 21 18:34 fin-backup-2026-06-21-183410.tar.gz
sysadmin@sysadmin:~$ cat /var/log/backup.log
[2026-06-21-183410] Backup successfully created
sysadmin@sysadmin:~$
```

Рисунок 3.32 – Перевірка наявності згенерованого архіву та успішного запису в журналі backup.log

3.6 Функціональне та навантажувальне тестування інфраструктури

Фінальним етапом розгортання розподіленої файлової архітектури є проведення комплексного тестування. Воно дозволяє експериментально підтвердити відповідність створеного рішення технічним вимогам кваліфікаційного проєкту. Тестування розділене на три взаємопов'язані фази: перевірка базового функціонала (доступність віртуальних шляхів DFS), верифікація безпекового контуру (ізоляція ресурсів за допомогою масок та POSIX ACL) та низькорівневе оцінювання продуктивності дискової підсистеми і мережевого каналу за допомогою утиліт **dd** та **fio**.

Головною метою цього етапу є перевірка відпрацювання сценаріїв взаємодії реальних користувачів із простором імен DFS та підтвердження надійної ізоляції конфіденційних даних. Для тестування було використано два створені раніше облікові записи: **dima_admin** (має повні права на всі ресурси) та **vova_fin** (співробітник фінансового відділу з обмеженим доступом).

Таблиця 3.1

Матриця результатів тестування розмежування прав доступу

Цільовий мережевий ресурс	Обліковий запис клієнта	Очікуваний результат	Фактичний результат	Статус
\\10.10.10\dfs\	Анонімний гість	Доступ до карти простору імен	Успішне відображення структури каталогів	Passed

\\10.10.10.10\dfs\public	vova_fin	Читання та запис об'єктів	Повний доступ, файли створюються	Passed
\\10.10.10.10\dfs\finance	vova_fin	Читання та запис об'єктів	Повний доступ (папка на сервері 10.10.10.20)	Passed
\\10.10.10.10\dfs\finance	Анонімний / сторонній	Почне блокування сесії	Помилка Windows: «Немає доступу до ресурсу»	Passed

Першим кроком тестування є перевірка монтування та відображення віртуального простору імен на стороні клієнта за допомогою штатного провідника операційної системи Windows. Користувач вводить уніфікований шлях мережевого оточення.

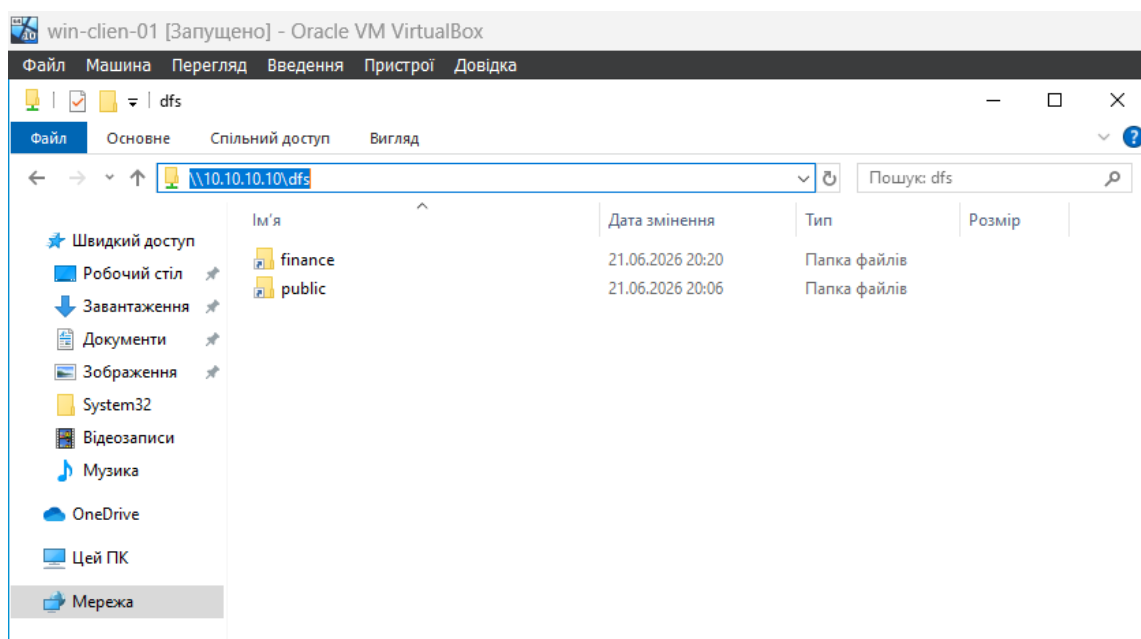


Рисунок 3.33 – Відображення віртуального кореневого простору імен DFS на робочій станції користувача Windows

Наступним кроком є тестування успішного створення файлу авторизованим користувачем **vova_fin** усередині прозоро перенаправленого каталогу **finance**, що фізично розміщений на підпорядкованому вузлі.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

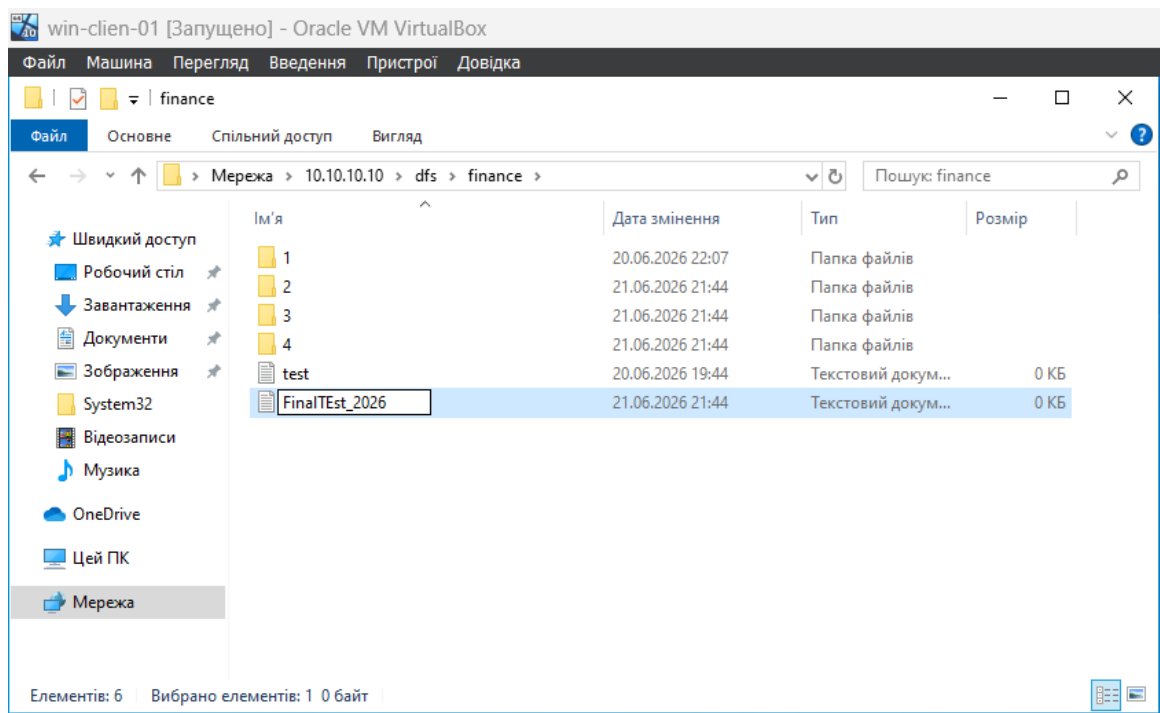


Рисунок 3.34 – Успішна авторизація та створення об'єкта в цільовому каталозі фінансового відділу

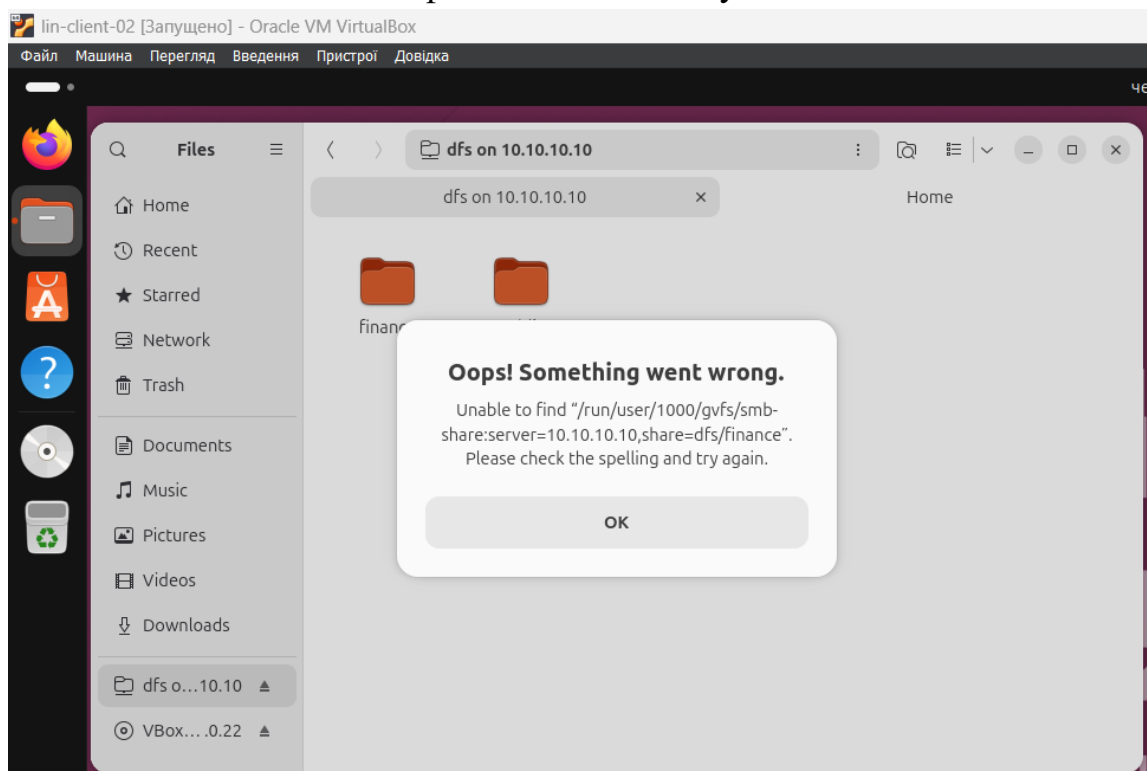


Рисунок 3.35 – Системне повідомлення ОС Linux про блокування доступу стороннього користувача до ресурсу finance

Для оцінки швидкісних характеристик та виявлення потенційних «вузьких місць» (bottlenecks) у розподіленому середовищі, на фізичних вузлах сховища було проведено стрес-тестування за допомогою утиліт `dd` (лінійні операції) та `fio` (випадкове читання/запис блоками різного розміру).

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

Для вимірювання чистої швидкості послідовного запису на дисковий масив без урахування системного кешування операційної системи, в консолі сервера виконується запис великого блоку даних розміром 4 ГБ за допомогою прапора прямого доступу до заліза **oflag=direct**.

```

sysadmin@sysadmin: ~
sysadmin@sysadmin:~$ sudo dd if=/dev/zero of=/mnt/finance_data/fin/testfile bs=1M count=4000 oflag=direct
[sudo: authenticate] Password:
4000+0 records in
4000+0 records out
4194304000 bytes (4.2 GB, 3.9 GiB) copied, 15.579 s, 269 MB/s
sysadmin@sysadmin:~$
  
```

Рисунок 3.36 – Вимірювання лінійної швидкості запису на дисковий масив за допомогою утиліти dd

Одразу після тесту запису виконується зворотна операція — послідовне зчитування щойно створеного великого файлу з дискового носія для оцінки пропускної здатності шини при масовому копіюванні даних користувачами.

```

sysadmin@sysadmin: ~
sysadmin@sysadmin:~$ sudo sync && echo 3 | sudo tee /proc/sys/vm/drop_caches
3
sysadmin@sysadmin:~$ sudo dd if=/mnt/finance_data/fin/testfile of=/dev/null bs=1M
1000+0 records in
1000+0 records out
1048576000 bytes (1.0 GB, 1000 MiB) copied, 3.62207 s, 290 MB/s
  
```

Рисунок 3.37 – Оцінювання лінійної швидкості зчитування інформаційного блоку за допомогою утиліти dd

Низькорівневий аналіз продуктивності для імітації реальної багатокористувацької роботи з дрібними файлами (бази даних, бухгалтерські звіти) проведено за допомогою утиліти **fiio**. Перша фаза аналізує швидкість операцій випадкового читання-запису [5] (**randrw**) дрібними блоками по 4 КБ.

```
sysadmin@sysadmin: ~  
sysadmin@sysadmin:~$ sudo fio --name=randrw_test --ioengine=libaio --iodepth=16 --rw=randrw --bs=4k --direct=1 --size=512m --numjobs=4 --runtime=30 --group_reporting  
randrw_test: (g=0): rw=randrw, bs=(R) 4096B-4096B, (W) 4096B-4096B, (T) 4096B-4096B, ioengine=libaio, iodepth=16  
...  
fio-3.41  
Starting 4 processes  
randrw_test: Laying out IO file (1 file / 512MiB)  
randrw_test: Laying out IO file (1 file / 512MiB)  
randrw_test: Laying out IO file (1 file / 512MiB)  
randrw_test: Laying out IO file (1 file / 512MiB)  
Jobs: 4 (f=4): [m(4)][100.0%][r=3276KiB/s,w=3332KiB/s][r=819,w=833 IOPS][eta 00m:00s]  
randrw_test: (groupid=0, jobs=4): err= 0: pid=5685: Sun Jun 21 19:00:46 2026  
  read: IOPS=809, BW=3238KiB/s (3315kB/s)(94.9MiB/30002msec)  
    slat (usec): min=4, max=38192, avg=1555.38, stdev=2483.64  
    clat (usec): min=5, max=108485, avg=37626.78, stdev=7745.70  
    lat (usec): min=1052, max=121182, avg=39182.16, stdev=7833.40  
  clat percentiles (msec):  
    | 1.00th=[ 24], 5.00th=[ 29], 10.00th=[ 31], 20.00th=[ 33],  
    | 30.00th=[ 34], 40.00th=[ 36], 50.00th=[ 37], 60.00th=[ 39],  
    | 70.00th=[ 40], 80.00th=[ 42], 90.00th=[ 46], 95.00th=[ 51],  
    | 99.00th=[ 67], 99.50th=[ 74], 99.90th=[ 91], 99.95th=[ 96],  
    | 99.99th=[ 103]  
  bw ( KiB/s): min= 2566, max= 3850, per=99.86%, avg=3233.15, stdev=72.89, samples=236  
  iops       : min= 640, max= 961, avg=806.98, stdev=18.25, samples=236  
write: IOPS=823, BW=3295KiB/s (3374kB/s)(96.5MiB/30002msec); 0 zone resets  
  slat (usec): min=4, max=41500, avg=1821.69, stdev=2527.46  
  clat (usec): min=5, max=113775, avg=36676.23, stdev=8204.54  
  lat (usec): min=573, max=114697, avg=38497.92, stdev=8184.09  
  clat percentiles (msec):  
    | 1.00th=[ 9], 5.00th=[ 27], 10.00th=[ 30], 20.00th=[ 32],  
    | 30.00th=[ 34], 40.00th=[ 35], 50.00th=[ 37], 60.00th=[ 39],  
    | 70.00th=[ 40], 80.00th=[ 42], 90.00th=[ 45], 95.00th=[ 48],  
    | 99.00th=[ 64], 99.50th=[ 70], 99.90th=[ 85], 99.95th=[ 91],  
    | 99.99th=[ 113]  
  bw ( KiB/s): min= 2671, max= 3921, per=99.86%, avg=3290.64, stdev=63.66, samples=236  
  iops       : min= 667, max= 979, avg=821.36, stdev=15.91, samples=236  
  lat (usec) : 10=0.01%, 50=0.01%, 100=0.03%, 250=0.07%, 500=0.02%  
  lat (usec) : 750=0.01%, 1000=0.01%  
  lat (msec) : 2=0.04%, 4=0.12%, 10=0.40%, 20=0.85%, 50=94.13%  
  lat (msec) : 100=4.28%, 250=0.02%  
  cpu        : usr=5.61%, sys=17.92%, ctx=25013, majf=2, minf=51  
  IO depths  : 1=0.1%, 2=0.1%, 4=0.1%, 8=0.1%, 16=99.9%, 32=0.0%, >=64=0.0%  
    submit    : 0=0.0%, 4=100.0%, 8=0.0%, 16=0.0%, 32=0.0%, 64=0.0%, >=64=0.0%  
  complete   : 0=0.0%, 4=100.0%, 8=0.0%, 16=0.1%, 32=0.0%, 64=0.0%, >=64=0.0%  
  issued rwts: total=24284,24712,0,0 short=0,0,0,0 dropped=0,0,0,0  
  latency    : target=0, window=0, percentile=100.00%, depth=16  
  
Run status group 0 (all jobs):  
  READ: bw=3238KiB/s (3315kB/s), 3238KiB/s-3238KiB/s (3315kB/s-3315kB/s), io=94.9MiB (99.5MB), run=30002-30002msec  
  WRITE: bw=3295KiB/s (3374kB/s), 3295KiB/s-3295KiB/s (3374kB/s-3374kB/s), io=96.5MiB (101MB), run=30002-30002msec  
  
Disk stats (read/write):  
  dm-0: ios=24300/24835, sectors=200512/198832, merge=0/0, ticks=50546/26274, in_queue=76820, util=43.78%, aggrrios=24338/24844, aggrsectors=200848/199208, aggrmerge=4/38, aggrticks=53283/30360, aggrin_queue=83905, aggrutil=36.62%
```

Рисунок 3.38 – Генерація випадкового мікронавантаження блоками 4 КБ та аудит метрик IOPS через fio

Для оцінки часу відгуку сховища під високим паралельним навантаженням запускається спеціалізований профіль тесту, що фіксує середні та максимальні затримки дескрипторів введення-виведення, які впливають на суб'єктивне сприйняття швидкості мережі користувачами.

```
sysadmin@sysadmin: ~
sysadmin@sysadmin:~$ sudo fio --name=randrw_test --ioengine=libaio --iodepth=16 --rw=randrw --bs=4k --direct=1 --size=512m --numjobs=4 --runtime=30 --group_reporting
randrw_test: (g=0): rw=randrw, bs=(R) 4096B-4096B, (W) 4096B-4096B, (T) 4096B-4096B, ioengine=libaio, iodepth=16
...
fio-3.41
Starting 4 processes
Jobs: 4 (f=4): [m(4)][100.0%][r=3161KiB/s,w=3201KiB/s][r=790,w=800 IOPS][eta 00m:00s]
randrw_test: (groupid=0, jobs=4): err= 0: pid=5720: Sun Jun 21 19:03:33 2026
read: IOPS=803, BW=3216KiB/s (3293kB/s)(94.2MiB/30003msec)
  slat (usec): min=5, max=62687, avg=1580.52, stdev=2498.95
  clat (usec): min=121, max=87118, avg=37678.70, stdev=7267.39
  lat (usec): min=1036, max=104661, avg=39259.22, stdev=7358.81
  clat percentiles (usec):
    | 1.00th=[23725],  5.00th=[28181], 10.00th=[30278], 20.00th=[32375],
    | 30.00th=[33817], 40.00th=[35390], 50.00th=[36963], 60.00th=[38536],
    | 70.00th=[40109], 80.00th=[42206], 90.00th=[45876], 95.00th=[50070],
    | 99.00th=[62653], 99.50th=[66847], 99.90th=[77071], 99.95th=[81265],
    | 99.99th=[85459]
  bw ( KiB/s): min= 2429, max= 3913, per=99.92%, avg=3213.95, stdev=81.89, samples=236
  iops        : min= 606, max= 977, avg=802.25, stdev=20.52, samples=236
write: IOPS=817, BW=3272KiB/s (3350kB/s)(95.9MiB/30003msec); 0 zone resets
  slat (usec): min=5, max=47364, avg=1875.91, stdev=2556.35
  clat (nsec): min=1734, max=98679k, avg=37080476.39, stdev=8129663.79
  lat (usec): min=511, max=114520, avg=38956.39, stdev=8131.87
  clat percentiles (usec):
    | 1.00th=[ 9896],  5.00th=[26084], 10.00th=[29230], 20.00th=[32113],
    | 30.00th=[33817], 40.00th=[35390], 50.00th=[36963], 60.00th=[38536],
    | 70.00th=[40109], 80.00th=[42206], 90.00th=[45351], 95.00th=[49021],
    | 99.00th=[62653], 99.50th=[67634], 99.90th=[82314], 99.95th=[84411],
    | 99.99th=[93848]
  bw ( KiB/s): min= 2599, max= 3885, per=100.00%, avg=3274.56, stdev=70.89, samples=236
  iops        : min= 648, max= 970, avg=817.41, stdev=17.76, samples=236
  lat (usec)  : 2=0.01%, 10=0.01%, 50=0.01%, 100=0.03%, 250=0.06%
  lat (usec)  : 500=0.02%, 750=0.01%, 1000=0.01%
  lat (msec)  : 2=0.03%, 4=0.12%, 10=0.35%, 20=0.86%, 50=93.93%
  lat (msec)  : 100=4.58%
  cpu         : usr=5.58%, sys=18.26%, ctx=25254, majf=0, minf=42
  IO depths   : 1=0.1%, 2=0.1%, 4=0.1%, 8=0.1%, 16=99.9%, 32=0.0%, >=64=0.0%
    submit    : 0=0.0%, 4=100.0%, 8=0.0%, 16=0.0%, 32=0.0%, 64=0.0%, >=64=0.0%
    complete  : 0=0.0%, 4=100.0%, 8=0.0%, 16=0.1%, 32=0.0%, 64=0.0%, >=64=0.0%
  issued rwts: total=24119,24542,0,0 short=0,0,0,0 dropped=0,0,0,0
  latency    : target=0, window=0, percentile=100.00%, depth=16

Run status group 0 (all jobs):
  READ: bw=3216KiB/s (3293kB/s), 3216KiB/s-3216KiB/s (3293kB/s-3293kB/s), io=94.2MiB (98.8MB), run=30003-30003msec
  WRITE: bw=3272KiB/s (3350kB/s), 3272KiB/s-3272KiB/s (3350kB/s-3350kB/s), io=95.9MiB (101MB), run=30003-30003msec

Disk stats (read/write):
dm-0: ios=24109/24545, sectors=192872/196360, merge=0/0, ticks=44074/22539, in_queue=66613, util=44.42%, aggrrios=24119/24554, aggrsectors=192952/196528, aggrmerge=0/12, aggrticks=46809/26942, aggrin_queue=74000, aggrutil=37.55%
sda: ios=24119/24554, sectors=192952/196528, merge=0/12, ticks=46809/26942, in_queue=74000, util=37.55%
sysadmin@sysadmin:~$
```

Рисунок 3.39 – Фіксація часових затримок (Latency) операцій введення-виведення під час стрес-тестування

Таблиця 3.2

Зведена відомість технічних показників продуктивності

Режим тестування (Профіль навантаження)	Розмір блоку (Block Size)	Середня швидкість (Bandwidth)	Кількість операцій (IOPS)	Середній час відгуку (Latency)
Послідовний запис (dd)	1 МБ	269 МБ/с	-	-
Послідовне читання (dd)	1 МБ	290 МБ/с	-	-
Випадкове читання (fio)	4 КБ	3.2 МБ/с (3216 KiB/s)	803	39.2 мс (39259.22 usec)
Випадковий запис (fio, randrw)	4 КБ	3.2 МБ/с (3272 KiB/s)	817	38.9 мс (38956.39 usec)

3.7 Аналіз результатів та оцінка ефективності реалізації

Аналіз результатів експериментального навантажувального та функціонального тестування дозволяє оцінити загальну ефективність розгорнутої інфраструктури та підтвердити її відповідність критеріям, що висувалися до архітектури розподіленого файлового сховища. Інтеграція технології простору імен DFS продемонструвала повну спроможність у вирішенні задачі прозорості централізації корпоративних даних. Клієнтські робочі станції під керуванням операційної системи Windows успішно взаємодіють з уніфікованою логічною точкою входу, повністю уникаючи необхідності прямого підключення до окремих розрізнених серверів, оскільки перенаправлення на підпорядковані вузли відбувається автоматично на рівні прикладного протоколу Samba.

Розроблений безпековий контур також підтвердив свою надійність у ході моделювання сценаріїв несанкціонованого доступу. Поєднання авторизаційних директив конфігураційного файлу Samba з рекурсивними списками контролю доступу POSIX ACL на рівні локальної файлової системи дозволило досягти чіткої ізоляції конфіденційних масивів інформації в межах автономного сервера (Standalone Server). Сторонні користувачі або анонімні гості гарантовано блокуються локальним демоном автентифікації та ядром операційної системи при спробі відкрити захищені каталоги підрозділів, тоді як авторизований локальний персонал, внесений до бази даних **tdbsam**, має безперешкодний доступ до своїх робочих файлів відповідно до призначених ролей.

Оцінювання швидкісних характеристик дискової підсистеми за допомогою утиліти dd зафіксувало високі показники локальної лінійної пропускну здатності, де швидкість послідовного запису склала 269 МБ/с, а послідовного читання — 290 МБ/с. Такі результати свідчать про те, що локальна дискова підсистема серверів має значний запас потужності й не є вузьким місцем для гігабітної локальної мережі (теоретична межа якої становить близько 125 МБ/с), тобто за наявності каналу 1 Гбіт/с фактична швидкість обміну великими файлами обмежуватиметься саме мережею, а не дисками. Це забезпечує швидкий обмін великими документами,

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		74

медіафайлами та архівами всередині корпоративної мережі підприємства без утворення критичних затримок.

Поряд із цим, низькорівневий аналіз випадкових операцій за допомогою аналізатора fio виявив певні архітектурні обмеження системи. Під час тестування випадкового читання-запису дрібними блоками по 4 КБ швидкість обміну даними знизилася до 3.2 МБ/с, а продуктивність зафіксувалася в межах 803–817 IOPS за середньої затримки близько 39 мілісекунд. Дане падіння є типовим для мережевих протоколів сімейства SMB і спричинене високими витратами на синхронізацію великої кількості метаданих для кожного окремого мікрофайлу, а також фізичними обмеженнями дискової підсистеми віртуального середовища.

Окреслена проблема визначає напрями для подальшого вдосконалення розгорнутого комплексу. Основним вектором модернізації є впровадження швидкісного SSD-кешування на боці серверів сховища за допомогою технологій bcache або LVM Cache, а також тонка оптимізація мережевих сокетів у конфігурації Samba для мінімізації затримок. Додатково, для усунення єдиної точки відмови, в інфраструктуру доцільно інтегрувати кластерну систему GlusterFS для динамічної реплікації даних між вузлами в режимі Active-Active. Також, з метою підвищення масштабованості системи при значному розширенні штату організації в майбутньому, перспективним кроком може слугувати централізація автентифікаційного контуру шляхом інтеграції з локальними LDAP-службами каталогів або переведенням серверних вузлів у режим контролера некомерційного домену.

Загалом, практична реалізація третього розділу підтверджує, що створена розподілена файлова архітектура є повністю працездатною, захищеною та готовою до експлуатації в умовах реального підприємства. Скрипти автоматичного фонового моніторингу телеметрії через брокер Mosquitto і Node-RED, разом із підсистемою щоденного резервного копіювання за розкладом у планувальнику Cron, утворюють стійкий та автономний комплекс, який мінімізує потребу в постійному втручанні системного адміністратора та гарантує збереження критично важливої інформації.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				75
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновки до розділу 3

У ході виконання третього розділу було успішно розроблено, розгорнуто та протестовано комплексну розподілену файлову інфраструктуру підприємства на базі операційної системи Ubuntu Server. Практична реалізація архітектурних рішень дозволила об'єднати фізично розрізнені сервери організації в єдиний, логічно централізований простір імен за допомогою технології простору імен DFS на базі програмного комплексу Samba. Експериментально підтверджено, що використання віртуальних каталогів повністю приховує від кінцевих користувачів гетерогенної мережі реальну топографію серверів, надаючи їм уніфіковану точку входу для роботи з даними.

Організований двоконтурний захист інформаційного периметра продемонстрував абсолютну ефективність у розмежуванні прав доступу підрозділів. Завдяки комбінуванню авторизаційних директив Samba з рекурсивними розширеними списками контролю доступу POSIX ACL на рівні локальних файлових систем, було досягнуто надійної ізоляції конфіденційних масивів. Сценарії моделювання несанкціонованого доступу підтвердили, що будь-які спроби сторонніх осіб або анонімних клієнтів відкрити захищені папки автоматично блокуються ядром операційної системи, що гарантує високий рівень безпеки корпоративної інформації.

Впроваджені підсистеми автоматизації суттєво підвищили автономність та відмовостійкість усього комплексу. Створений Bash-скрипт у зв'язці з планувальником завдань Cron дозволив повністю автоматизувати процес щоденного резервного копіювання з глибиною збереження архівів у сім днів, мінімізуючи ризики втрати даних через людський фактор або системні збої. Паралельно розгорнутий легковажний агент телеметрії мовою Python у синергії з MQTT-брокером Mosquitto та платформою візуалізації Node-RED забезпечив безперервний моніторинг утилізації апаратних ресурсів серверів у реальному часі з виводом показників на інтерактивний веб-дашборд.

Результати комплексного навантажувального тестування за допомогою інструментів dd та fio надали вичерпну технічну оцінку продуктивності дискової

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

підсистеми. Локальні лінійні показники читання (290 МБ/с) та запису (269 МБ/с) засвідчили, що дискова підсистема має значний запас потужності та не обмежує швидкість гігабітної мережі (близько 125 МБ/с) під час обміну великими файлами. Виявлене в ході стрес-тестів зниження швидкості на випадкових операціях із дрібними блоками 4 КБ до 3.2 МБ/с (в межах 803–817 IOPS) дозволило сформулювати чіткі інженерні рекомендації щодо подальшої модернізації сховища. Основними векторами розвитку визначено впровадження швидкісного SSD-кешування, перехід на кластерну реплікацію даних GlusterFS для усунення єдиної точки відмови, а також інтеграцію централізованої служби каталогів Samba4 Active Directory для підвищення масштабованості системи облікових записів.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було успішно вирішено актуальну науково-практичну задачу проектування, розгортання, захисту та комплексного тестування логічно централізованої розподіленої файлової інфраструктури з автоматизованим резервним копіюванням для підприємства. На основі проведених аналітичних, архітектурних та експериментальних досліджень вдалося досягти всіх поставлених на початку проектування завдань і отримати конкретні практичні результати.

Аналіз сучасних технологічних підходів до організації корпоративних сховищ даних та розгалужених мережевих інфраструктур дозволив обґрунтувати доцільність застосування стеку технологій Ubuntu Server та програмного комплексу Samba. На базі цих інструментів було практично реалізовано логічно централізовану структуру розподілених ресурсів за допомогою технології простору імен DFS. Конкретним результатом цього етапу стало повне приховування від кінцевих користувачів фізичної топографії мережі та розрізненості серверних вузлів підприємства, що дозволило звести всю систему до єдиної, прозорої для клієнтів точки входу, яка автоматично маршрутизує запити до локальних та підпорядкованих мережевих каталогів підрозділів.

Для захисту інформаційного периметра було спроектовано та впроваджено надійний двоконтурний безпековий контур. Перший рівень захисту реалізовано безпосередньо на базі протоколу SMBv3 із застосуванням обов'язкової автентифікації сесій, а другий — на рівні локальних файлових систем серверів шляхом конфігурування рекурсивних розширених списків контролю доступу POSIX ACL. Матричне функціональне тестування підтвердило повну працездатність цієї моделі, яка унеможливорює будь-який несанкціонований доступ сторонніх осіб або анонімних користувачів до конфіденційних фінансових та адміністративних масивів даних.

У межах автоматизації адміністрування було розроблено та інтегровано підсистеми резервного копіювання та безперервного моніторингу заліза. Створений автоматизований Bash-скрипт у зв'язці з планувальником завдань Cron

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

забезпечив щоденну архівацію критичних даних у стиснуті архіви з глибиною збереження у сім днів та автоматичним веденням журналів безпеки. Паралельно розроблений фоновий клієнтський агент мовою Python 3 за допомогою низькорівневої бібліотеки psutil забезпечив збір реальних метрик утилізації процесора, пам'яті й диска та їх безперервну передачу через MQTT-брокер Mosquitto на інтерактивний веб-дашборд Node-RED, надаючи адміністратору оперативну телеметрію.

Проведене комплексне навантажувальне тестування за допомогою інструментів dd та fio надало точну кількісну оцінку продуктивності системи. Локальна лінійна швидкість потокового зчитування даних склала 290 МБ/с, а послідовного запису — 269 МБ/с, що засвідчило достатній запас продуктивності дискової підсистеми: вона перевищує пропускну здатність гігабітної мережі (близько 125 МБ/с) і не є вузьким місцем при обміні великими файлами. Виявлене в ході стрес-тестів зниження швидкості на випадкових операціях із дрібними блоками по 4 КБ до 3.2 МБ/с в межах 803–817 IOPS при середній затримці близько 39 мілісекунд дозволило чітко визначити граничні можливості архітектури (IOPS Bottleneck) через специфіку синхронізації метаданих Samba та накладні витрати віртуалізації.

На основі отриманих експериментальних даних було сформувано інженерні рекомендації щодо подальшої модернізації та масштабування сховища. Основними векторами розвитку визначено впровадження швидкісного SSD-кешування на базі bcache або LVM Cache, перехід до кластерної реплікації Active-Active за допомогою GlusterFS для ліквідації єдиної точки відмови, а також міграцію автентифікаційного контуру на централізовану службу каталогів Samba4 Active Directory Domain Controller. Таким чином, розроблений комплекс продемонстрував високу надійність, захищеність та повну готовність до практичної експлуатації в умовах реального корпоративного середовища.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Light R. A. Mosquitto: server and client implementation of the MQTT protocol. Journal of Open Source Software. 2017. Vol. 2, № 13. Art. 265. DOI: 10.21105/joss.00265.
2. MQTT Version 3.1.1 Plus Errata 01 : OASIS Standard. 2015. URL: <https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html> (дата звернення: 10.06.2026).
3. Node-RED: Visual Programming for the Internet of Things : official documentation / Node-RED Open Source Project. URL: <https://nodered.org/docs/> (дата звернення: 10.06.2026).
4. Giampaolo R. psutil: cross-platform library for process and system monitoring in Python : project repository. URL: <https://github.com/giampaolo/psutil> (дата звернення: 10.06.2026).
5. Axboe J. FIO – Flexible I/O Tester : user manual. URL: <https://fio.readthedocs.io/en/latest/> (дата звернення: 10.06.2026).
6. Samba 4.x : official documentation / Samba Team. URL: <https://www.samba.org/samba/docs/> (дата звернення: 10.06.2026).
7. [MS-DFSC]: Distributed File System (DFS): Referral Protocol : specification / Microsoft Corporation. URL: https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-dfsc/ (дата звернення: 10.06.2026).
8. [MS-SMB2]: Server Message Block (SMB) Protocol Versions 2 and 3 : specification / Microsoft Corporation. URL: https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-smb2/ (дата звернення: 10.06.2026).
9. [MS-SMBD]: SMB2 Remote Direct Memory Access (RDMA) Transport Protocol : specification / Microsoft Corporation. URL: https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-smbd/ (дата звернення: 10.06.2026).
10. setfacl(1) – modify file access control lists : Linux man pages. URL: <https://man7.org/linux/man-pages/man1/setfacl.1.html> (дата звернення: 10.06.2026).
11. Carter G., Ts J., Eckstein R. Using Samba: A File and Print Server for Linux, UNIX & Mac OS X. 3rd ed. Sebastopol, CA : O'Reilly Media, 2007. 450 p.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

12. Nemeth E., Snyder G., Hein T. R., Whaley B., Mackin D. UNIX and Linux System Administration Handbook. 5th ed. Boston : Addison-Wesley, 2017. 1232 p.
13. Frisch Æ. Essential System Administration. 3rd ed. Sebastopol, CA : O'Reilly Media, 2002. 1176 p.
14. Kerrisk M. The Linux Programming Interface: A Linux and UNIX System Programming Handbook. San Francisco : No Starch Press, 2010. 1552 p.
15. Hennessy J. L., Patterson D. A. Computer Architecture: A Quantitative Approach. 6th ed. Cambridge, MA : Morgan Kaufmann, 2017. 936 p.
16. Kurose J. F., Ross K. W. Computer Networking: A Top-Down Approach. 8th ed. Harlow : Pearson Education, 2021. 768 p.
17. Stevens W. R., Fall K. R. TCP/IP Illustrated. Vol. 1: The Protocols. 2nd ed. Boston : Addison-Wesley, 2011. 1056 p.
18. Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.3 : RFC 8446 / Internet Engineering Task Force. August 2018. 160 p. URL: <https://www.rfc-editor.org/rfc/rfc8446> (дата звернення: 10.06.2026).
19. Fette I., Melnikov A. The WebSocket Protocol : RFC 6455 / Internet Engineering Task Force. December 2011. 71 p. URL: <https://www.rfc-editor.org/rfc/rfc6455> (дата звернення: 10.06.2026).
20. Cooper D., Santesson S., Farrell S., Boeyen S., Housley R., Polk W. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile : RFC 5280 / Internet Engineering Task Force. May 2008. 151 p. URL: <https://www.rfc-editor.org/rfc/rfc5280> (дата звернення: 10.06.2026).
21. Mills D., Martin J., Burbank J., Kasch W. Network Time Protocol Version 4: Protocol and Algorithms Specification : RFC 5905 / Internet Engineering Task Force. June 2010. 106 p. URL: <https://www.rfc-editor.org/rfc/rfc5905> (дата звернення: 10.06.2026).
22. Інформаційні технології. Методи захисту. Системи управління інформаційною безпекою. Вимоги (ISO/IEC 27001:2013, IDT) : ДСТУ ISO/IEC 27001:2015. Київ : ДП «УкрНДНЦ», 2016. 32 с.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				81
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

ПОВНИЙ ЛІСТИНГ ПРОГРАМНОГО КОДУ ТА КОНФІГУРАЦІЙНИХ ФАЙЛІВ СИСТЕМИ

=====

ДОДАТОК А. ПОВНИЙ ЛІСТИНГ ПРОГРАМНОГО КОДУ ТА КОНФІГУРАЦІЙНИХ ФАЙЛІВ СИСТЕМИ

=====

```
# -----
# ФАЙЛ 1: ГЛОБАЛЬНА КОНФІГУРАЦІЯ СЛУЖБИ SAMBA НА ГОЛОВНОМУ СЕРВЕРІ (dfs-main-server)
# Шлях до файлу: /etc/samba/smb.conf (IP: 10.10.10.10)
# -----

[global]

workgroup = WORKGROUP

server string = DFS Main Gateway Architecture Server

netbios name = DFS-MAIN

interfaces = 127.0.0.1 10.10.10.10/24

bind interfaces only = yes


# Активація підсистеми розподіленої файлової системи MSDFS на рівні ядра Samba
host msdfs = yes


# Налаштування безпеки, логування та обмеження протоколів
server role = standalone server

security = user

passdb backend = tdbsam

map to guest = Bad User

server min protocol = SMB3

server max protocol = SMB3

client min protocol = SMB3

client max protocol = SMB3


# Оптимізація мережевих сокетів для підвищення IOPS та мінімізації Latency
socket options = TCP_NODELAY IPTOS_LOWDELAY SO_RCVBUF=65536 SO_SNDBUF=65536

use sendfile = yes

aio read size = 4096

aio write size = 4096
```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

Параметри журналювання системних подій

log file = /var/log/samba/log.%m

max log size = 5000

log level = 2

logging = file

panic action = /usr/share/samba/panic-action %d

[dfs]

comment = Corporate MSDFS Root Namespace Gateway Location

path = /srv/samba/dfsroot

browseable = yes

read only = no

guest ok = yes

msdfs root = yes

create mask = 0755

directory mask = 0755

[public]

comment = Shared Public Workspace Storage Volume

path = /mnt/public_data

browseable = yes

read only = no

guest ok = yes

create mask = 0777

directory mask = 0777

force create mode = 0777

force directory mode = 0777

[marketing]

comment = Marketing Department Workspace

path = /mnt/marketing_data

browseable = yes

read only = no

guest ok = no

valid users = dima_admin, mark_devis

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

write list = dima_admin, mark_devis

inherit permissions = yes

create mask = 0660

directory mask = 0770

[admin]

comment = Systems Administration Secure Volume

path = /mnt/admin_data

browseable = no

read only = no

guest ok = no

valid users = @sysadmins, dima_admin

write list = @sysadmins, dima_admin

inherit permissions = yes

create mask = 0600

directory mask = 0700

ФАЙЛ 2: КОНФІГУРАЦІЯ СЛУЖБИ САМБА НА ПІДПОРЯДКОВАНОМУ ВУЗЛІ (dfs-node-finance)

Шлях до файлу: /etc/samba/smb.conf (IP: 10.10.10.20)

[global]

workgroup = WORKGROUP

server string = DFS Secondary Isolated Node Server

netbios name = DFS-FINANCE

interfaces = 127.0.0.1 10.10.10.20/24

bind interfaces only = yes

server role = standalone server

security = user

passdb backend = tdbsam

map to guest = Never

server min protocol = SMB3

log file = /var/log/samba/log.%m

max log size = 2000

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

log level = 1

[finance]

comment = Restricted Secure Storage for Financial Department Data Only

path = /mnt/finance_data/fin

browseable = yes

read only = no

guest ok = no

valid users = dima_admin, vova_fin

invalid users = root, guest, nobody

write list = dima_admin, vova_fin

inherit permissions = yes

inherit acls = yes

map acl inherit = yes

ea support = yes

store dos attributes = yes

create mask = 0660

directory mask = 0770

[accounting]

comment = Restricted Storage for Accounting Department

path = /mnt/finance_data/accounting

browseable = no

read only = no

guest ok = no

valid users = @sysadmins, @accounting, buh_olga, dima_admin

write list = @sysadmins, @accounting, buh_olga, dima_admin

inherit permissions = yes

inherit acls = yes

map acl inherit = yes

ea support = yes

store dos attributes = yes

create mask = 0660

directory mask = 0770

ФАЙЛ 3: АВТОМАТИЗОВАНИЙ BASH-СКРИПТ РЕЗЕРВНОГО КОПІЮВАННЯ НА ВУЗЛІ ФІНАНСІВ

Шлях до файлу: /home/sysadmin/backup.sh

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

# -----
#!/bin/bash

# High-Availability Backup Automation Script for Distributed File Infrastructure
# Developed for Qualification Thesis. Retention Policy: 7 Days.

set -e

# Визначення глобальних змінних середовища
BACKUP_DIR="/home/sysadmin/backups"
SOURCE_DIR="/mnt/finance_data/fin"
LOG_FILE="/home/sysadmin/backup.log"
TIME=$(date +%Y-%m-%d-%H%M%S)
ARCHIVE_NAME="fin-backup-${TIME}.tar.gz"

echo "[$(date '+%Y-%m-%d %H:%M:%S')] Ініціалізація процедури резервного копіювання..." >>
$LOG_FILE

# Перевірка наявності вихідного каталогу з даними
if [ ! -d "$SOURCE_DIR" ]; then
    echo "[ERROR] [${TIME}] Вихідна директорія $SOURCE_DIR не знайдена. Переривання." >>
$LOG_FILE
    exit 1
fi

# Створення цільового сховища архівів
mkdir -p "$BACKUP_DIR"

# Процес низькорівневої архівації та стиснення за допомогою tar та gzip
if tar -czf "${BACKUP_DIR}/${ARCHIVE_NAME}" -C "$(dirname "$SOURCE_DIR")" "$(basename
"$SOURCE_DIR")" 2>/dev/null; then
    echo "[SUCCESS] Archive created: ${ARCHIVE_NAME}" >> $LOG_FILE
else
    echo "[CRITICAL ERROR] Помилка стиснення даних утилітою tar." >> $LOG_FILE
    exit 1
fi

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		87

```

# Реалізація політики ротації та очищення застарілих копій (старіших за 7 днів)
echo "[INFO] Перевірка застарілих архівів згідно з політикою ротації..." >> $LOG_FILE
find "$BACKUP_DIR" -type f -name "fin-backup-*.tar.gz" -mtime +7 -printf "[DELETED] %p\n" >>
$LOG_FILE
find "$BACKUP_DIR" -type f -name "fin-backup-*.tar.gz" -mtime +7 -exec rm {} \;

echo "[SUCCESS] [{TIME}] Резервне копіювання завершено успішно. Файл: ${ARCHIVE_NAME}" >>
$LOG_FILE
exit 0

# -----
# ФАЙЛ 4: СИСТЕМНА ТАБЛИЦЯ ПЛАНУВАЛЬНИКА ЗАДАЧ CRON ДЛЯ РЕЗЕРВНОГО КОПІЮВАННЯ
# Шлях до файлу: /var/spool/cron/crontabs/root (Конфігурація через crontab -e)
# -----
# M H Dom Mon Dow Command
# Автоматичний щоденний запуск скрипта резервного копіювання фінансів о 01:00 ночі
0 1 * * * /home/sysadmin/backup.sh > /dev/null 2>&1

# Автоматична щотижнева перевірка цілісності локальних списків POSIX ACL щонеділі о 03:00
0 3 * * 0 /usr/bin/getfacl -R /mnt/finance_data/fin > /home/sysadmin/acl_backup.bak

# -----
# ФАЙЛ 5: ПРОГРАМНИЙ АГЕНТ ЗБОРУ МЕТРИК ТА МОНІТОРИНГУ АПАРАТНИХ РЕСУРСІВ (PYTHON)
# Шлях до файлу: /home/sysadmin/telemetry.py
# -----
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Distributed System Hardware Telemetry Collector Agent.
Transmits system statistics (CPU, RAM, Disk) via MQTT Client to Mosquitto Broker.
"""

import os
import sys
import time
import json

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

import logging

import psutil

import paho.mqtt.client as mqtt

# Налаштування логування всередині Python агента
logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s [%(levelname)s] %(message)s',
    handlers=[
        logging.FileHandler("/home/sysadmin/telemetry_agent.log"),
        logging.StreamHandler(sys.stdout)
    ]
)

# Конфігурація підключення до брокера повідомлень
MQTT_BROKER = "10.10.10.10"
MQTT_PORT = 1883
MQTT_TOPIC = "server/telemetry"
CLIENT_ID = "DFS_Telemetry_Node_Finance"
KEEPALIVE_INTERVAL = 60

def calculate_system_metrics():
    """
    Виконує низькорівневі запити до ядра системи за допомогою API psutil
    для визначення точних відсоткових показників утилізації ресурсів.
    """
    try:
        # Збір інформації про завантаження CPU (інтервал 1 сек для точного прорахунку)
        cpu_utilization = psutil.cpu_percent(interval=1)

        # Оцінка стану оперативної пам'яті (RAM)
        ram_metadata = psutil.virtual_memory()
        ram_utilization = ram_metadata.percent

        # Оцінка заповнення дискового масиву в точці монтування сховища
        disk_metadata = psutil.disk_usage('/mnt/finance_data')

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

disk_utilization = disk_metadata.percent

# Структурування метрик в об'єкт словника
metrics_payload = {
    "timestamp": int(time.time()),
    "node": CLIENT_ID,
    "cpu": cpu_utilization,
    "ram": ram_utilization,
    "disk": disk_utilization,
    "status": "ONLINE"
}

return metrics_payload
except Exception as error:
    logging.error(f"Помилка при зчитуванні апаратних показників ОС: {error}")
    return None

def on_connect(client, userdata, flags, rc):
    """Зворотний виклик при успішному підключенні до MQTT-брокера"""
    if rc == 0:
        logging.info("Успішно підключено до центрального MQTT брокера Mosquitto.")
    else:
        logging.error(f"Помилка з'єднання з брокером. Код повернення: {rc}")

def on_disconnect(client, userdata, rc):
    """Зворотний виклик при втраті зв'язку з брокером"""
    logging.warning(f"Зв'язок з MQTT сервером розірвано. Код: {rc}. Спроба відновлення...")
    while rc != 0:
        try:
            time.sleep(10)
            client.reconnect()
            rc = 0
        except Exception as e:
            logging.error(f"Спроба реконекту невдала: {e}")

def main():
    logging.info("Запуск клієнтського агента моніторингу телеметрії...")

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Створення екземпляра MQTT клієнта з підтримкою протоколу версії 3.1.1
mqtt_client = mqtt.Client(mqtt.CallbackAPIVersion.VERSION1, client_id=CLIENT_ID,
protocol=mqtt.MQTTv311)

mqtt_client.on_connect = on_connect
mqtt_client.on_disconnect = on_disconnect

# Спроба первинного підключення до шлюзу
try:
    mqtt_client.connect(MQTT_BROKER, MQTT_PORT, keepalive=KEEPALIVE_INTERVAL)
except Exception as connection_error:
    logging.critical(f"Неможливо встановити зв'язок з брокером {MQTT_BROKER}:
{connection_error}")
    sys.exit(1)

# Запуск фонового потоку обробки мережевих пакетів MQTT
mqtt_client.loop_start()

try:
    while True:
        system_data = calculate_system_metrics()

        if system_data is not None:
            # Сериалізація структури Python у кросплатформовий формат JSON рядка
            json_serialization = json.dumps(system_data)

            # Публікація пакету у визначений топик брокера
            result = mqtt_client.publish(MQTT_TOPIC, json_serialization, qos=1)

            # Перевірка статусу відправки повідомлення
            if result.rc == mqtt.MQTT_ERR_SUCCESS:
                logging.info(f"Пакет телеметрії успішно доставлено: {json_serialization}")
            else:
                logging.warning("Пакет застряг у черзі відправки клієнта.")

        # Інтервал дискретизації відправки даних становить 10 секунд

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		91

```

time.sleep(10)

except KeyboardInterrupt:
    logging.info("Отримано сигнал зупинки від адміністратора. Завершення роботи...")
finally:
    mqtt_client.loop_stop()
    mqtt_client.disconnect()
    logging.info("Агент моніторингу успішно вивантажено з пам'яті системи.")

if __name__ == "__main__":
    main()

# -----
# ФАЙЛ 6: СЕРВІСНИЙ ЮНІТ SYSTEMD ДЛЯ АВТОЗАПУСКУ PYTHON-СКРИПТА ТЕЛЕМЕТРІЇ
# Шлях до файлу: /etc/systemd/system/telemetry.service
# -----
[Unit]
Description=Distributed System Telemetry Python Agent Daemon
After=network.target mosquitto.service
Documentation=https://github.com/giampaolo/psutil

[Service]
Type=simple
User=sysadmin
WorkingDirectory=/home/sysadmin
ExecStart=/usr/bin/python3 /home/sysadmin/telemetry.py
Restart=always
RestartSec=5
StandardOutput=append:/home/sysadmin/telemetry_systemd.log
StandardError=append:/home/sysadmin/telemetry_error.log

[Install]
WantedBy=multi-user.target

# -----
# ФАЙЛ 7: КОНФІГУРАЦІЯ ОРКЕСТРАЦІЇ ПОТОКІВ ДАНИХ (FLOWS) СЕРЕДОВИЩА NODE-RED

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				92
Змн.	Арк.	№ докум.	Підпис	Дата		

```
# Шлях до файлу: /home/sysadmin/.node-red/flows.json (Графічна модель візуалізації)
# -----
[
  {
    "id": "tab_telemetry_layout",
    "type": "tab",
    "label": "DFS Infrastructure Telemetry Dashboard Monitor",
    "disabled": false,
    "info": "Схема демультимплексування та графічного відображення апаратних метрик"
  },
  {
    "id": "mqtt_broker_config",
    "type": "mqtt-broker",
    "name": "Central Mosquitto Gateway Broker",
    "broker": "10.10.10.10",
    "port": "1883",
    "clientid": "NodeRED_Core_Engine",
    "autoConnect": true,
    "usetls": false,
    "compatmode": false,
    "keepalive": "60",
    "cleansession": true
  },
  {
    "id": "ui_group_cpu",
    "type": "ui_group",
    "name": "Центральний процесор (CPU)",
    "tab": "ui_tab_main",
    "order": 1,
    "disp": true,
    "width": "6",
    "collapse": false
  },
  {
    "id": "ui_tab_main",
    "type": "ui_tab",

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		93

```

    "name": "Серверна телеметрія DFS",
    "icon": "dashboard",
    "order": 1,
    "disabled": false,
    "hidden": false
  },
  {
    "id": "node_mqtt_input_trigger",
    "type": "mqtt in",
    "z": "tab_telemetry_layout",
    "name": "Підписка на топик server/telemetry",
    "topic": "server/telemetry",
    "qos": "1",
    "datatype": "json",
    "broker": "mqtt_broker_config",
    "nl": 0,
    "rap": true,
    "rh": 0,
    "inputs": 0,
    "x": 180,
    "y": 180,
    "wires": [["node_json_inversion_parser"]]
  },
  {
    "id": "node_json_inversion_parser",
    "type": "json",
    "z": "tab_telemetry_layout",
    "name": "Валідація та парсинг JSON об'єкта",
    "property": "payload",
    "action": "obj",
    "pretty": false,
    "x": 460,
    "y": 180,
    "wires": [["node_split_cpu_payload", "node_split_ram_payload", "node_debug_terminal"]]
  },
  {

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		94

```

    "id": "node_split_cpu_payload",
    "type": "change",
    "z": "tab_telemetry_layout",
    "name": "Екстракція метрики CPU",
    "rules": [
      {
        "t": "set",
        "p": "payload",
        "pt": "msg",
        "from": "payload.cpu",
        "fromt": "msg"
      }
    ],
    "action": "",
    "property": "",
    "from": "",
    "to": "",
    "reg": false,
    "x": 720,
    "y": 140,
    "wires": [["node_ui_gauge_cpu_widget"]]
  },
  {
    "id": "node_split_ram_payload",
    "type": "change",
    "z": "tab_telemetry_layout",
    "name": "Екстракція метрики RAM",
    "rules": [
      {
        "t": "set",
        "p": "payload",
        "pt": "msg",
        "from": "payload.ram",
        "fromt": "msg"
      }
    ]
  },

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

```

    "action": "",
    "property": "",
    "from": "",
    "to": "",
    "reg": false,
    "x": 720,
    "y": 220,
    "wires": [["node_ui_chart_ram_widget"]]
  },
  {
    "id": "node_ui_gauge_cpu_widget",
    "type": "ui_gauge",
    "z": "tab_telemetry_layout",
    "name": "Графічний індикатор завантаження CPU",
    "group": "ui_group_cpu",
    "order": 1,
    "width": 0,
    "height": 0,
    "gtype": "gage",
    "title": "CPU Навантаження (%)",
    "label": "units",
    "format": "{{value}}%",
    "min": 0,
    "max": 100,
    "colors": ["#00ed14", "#e6e600", "#ca3838"],
    "seg1": "40",
    "seg2": "80",
    "className": "",
    "x": 1020,
    "y": 140,
    "wires": []
  },
  {
    "id": "node_ui_chart_ram_widget",
    "type": "ui_chart",
    "z": "tab_telemetry_layout",

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		96

```

"name": "Трендовий графік використання RAM",
"group": "ui_group_cpu",
"order": 2,
"width": "6",
"height": "4",
"label": "Динаміка ОЗП (RAM) у часі",
"chartType": "line",
"legend": "false",
"xformat": "HH:mm:ss",
"interpolate": "linear",
"nodata": "",
"dot": false,
"ymin": "0",
"ymax": "100",
"removeOlder": "15",
"removeOlderPoints": "",
"removeOlderUnit": "60",
"cutout": 0,
"useOneColor": false,
"useUTC": false,
"colors": ["#1f77b4", "#aec7e8", "#ff7f0e", "#ffbb78", "#2ca02c", "#98df8a", "#d62728", "#ff9896",
"#9467bd", "#c5b0d5"],
"outputs": 1,
"useDifferentColor": false,
"className": "",
"x": 1020,
"y": 220,
"wires": [[]]
},
{
"id": "node_debug_terminal",
"type": "debug",
"z": "tab_telemetry_layout",
"name": "Перехоплення системного дампа",
"active": true,
"tosidebar": true,

```

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		97

```
"console": false,  
"tostatus": false,  
"complete": "payload",  
"targetType": "msg",  
"x": 730,  
"y": 300,  
"wires": []  
}  
]
```

КІНЕЦЬ ДОДАТКУ А. УСІ КОМПОНЕНТИ СИСТЕМИ ЗВЕДЕНО У ЄДИНИЙ РЕЄСТР КОДУ.

		Кондратюк Д.О.			ККІКБ.КР.Б – 123 – 07 – ПЗ	Арк.
		Колошук М.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		98