МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВЕТЕРИНАРНОЇ МЕДИЦИНИ ТА БІОТЕХНОЛОГІЙ ІМ. С.З. ҐЖИЦЬКОГО

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

Другого (магістерського) рівня вищої освіти

на тему: “ Віртуалізація обчислювальних ресурсів для оптимізації роботи дата-центрів ”

Виконав: студент 6 курсу групи Іт-61
Спеціальності 126 – „Інформаційні системи та технології”
(шифр і назва)

Шаповал Роман Олександрович
(Прізвище та ініціали)

Керівник: к.т.н., доц. Падюка Р.І.
(Прізвище та ініціали)

Рецензенти: к.т.н., доц. Тимочко В.О.
(Прізвище та ініціали)

(Прізвище та ініціали)

ДУБЛЯНИ-2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВЕТЕРИНАРНОЇ МЕДИЦИНИ ТА БІОТЕХНОЛОГІЙ ІМ. С.З. ГЖИЦЬКОГО

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Другий (магістерський) рівень вищої освіти
Спеціальність 126 "Інформаційні системи та технології"

“ЗАТВЕРДЖУЮ”
Завідувач кафедри інформаційних технологій
д.т.н., проф. А.М. Тригуба
“____”_____2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу студенту

Шаповалу Роману Олександровичу

1. Тема роботи: «Віртуалізація обчислювальних ресурсів для оптимізації роботи дата-центрів»

Керівник роботи Падюка Роман Іванович, к.т.н., доцент.

Затверджені наказом по університету від 28 лютого 2025 року № 140/к-с.

2. Строк подання студентом роботи 05.12.2025 р.

3. Початкові дані до роботи: 1. Характеристики існуючої серверної інфраструктури; 2) Стан обчислювальних ресурсів і систем зберігання даних у дата-центрі; 3) Основи технологій віртуалізації та кластеризації обчислювальних систем; 4) Методи аналізу навантаження на серверні ресурси та показників їх ефективності; 5) Методика визначення економічної та енергетичної ефективності IT-інфраструктури.

4. Зміст розрахунково-пояснювальної записки:

1. Аналіз стану питання в практиці та теорії
2. Обґрунтування, вибір та реалізація інструментарію вирішення задачі
3. Результати вирішення задачі
4. Охорона праці та безпека в надзвичайних ситуаціях
5. Визначення ефективності впровадження інструментарію віртуалізації

Висновки та пропозиції.

Бібліографічний список.

Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень): Тема, автор, керівник магістерської роботи; Мета,завдання, об’єкт, предмет дослідження; Актуальність: чому потрібна віртуалізація в ДЦ; Архітектура віртуалізації: базова модель; Вимоги до інструментарію для оптимізації ДЦ;

Контейнеризація та оркестрація в оптимізації ДЦ; Реалізована інфраструктура: концепція кластера Proxmox; Етапи створення кластера (етапи GUI); Реалізація приєднання нод і параметри Join; Реалізація спільного сховища і High Availability, Перевірка відмовостійкості та кворуму; Моніторинг і результати вимірювань продуктивності; Ефективність і підсумкові висновки

6. Консультанти з розділів:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 2, 3, 5	*Падюка Р.І., доцент кафедри інформаційних технологій*		
4	*Городецький І.М., доцент кафедри інженерної механіки*		

7. Дата видачі завдання 03 березня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Приміт-ка
1	*Написання першого розділу та означення головних завдань роботи*	03.03.-01.04.25	
2	*Виконання другого розділу та формування головних показників для розрахунків*	02.04.-01.05.25	
3.	*Виконання третього розділу та формування початкових даних*	02.05.-01.06.25	
4.	*Виконання четвертого розділу та узагальнення отриманих результатів магістерської роботи*	02.06.-01.09.25	
5.	*Вартісне оцінення ефективності пропозицій роботи*	02.09.-01.10.25	
6.	*Завершення оформлення розрахунково-пояснювальної записки та аркушів графічної частини*	02.10.-01.11.25	
7.	*Завершення роботи в цілому*	02.11.-05.12.25	

Студент _____ Шаповал Р.О.
(підпис)

Керівник роботи _____ Падюка Р.І.
(підпис)

УДК 004.45:004.7:004.272

Віртуалізація обчислювальних ресурсів для оптимізації роботи дата-центрів – Шаповал Р.О. Магістерська робота. Кафедра ІТ. – Дубляни, Львівський НУВМБ ім. С.З. Гжицького, 2025.

82 с. текст. част., 30 рис., 1 табл., 14 арк. графічної частини, 20 літ. джерел

У роботі досліджено підходи до оптимізації роботи дата-центрів шляхом віртуалізації обчислювальних ресурсів. Проаналізовано сучасні платформи віртуалізації та обґрунтовано вибір Proxmox VE як інструментарію, що поєднує гіпервізор KVM, контейнеризацію, кластеризацію та засоби централізованого керування. Практично реалізовано кластер Proxmox VE, підключено спільне сховище та налаштовано High Availability, експериментально перевірено автоматичне відновлення сервісів при відмові вузлів. Проведено оцінювання режимів балансування й консолідації, побудовано графіки метрик CPU, I/O та p95 часу відповіді. Виконано розрахунок енергетичної, економічної ефективності та трудомісткості впровадження.

Ключові слова: віртуалізація; дата-центр; Proxmox VE; кластеризація; висока доступність; консолідація навантаження; моніторинг

Keywords: virtualization; data center; Proxmox VE; clustering; high availability; workload consolidation; monitoring

ЗМІСТ

ВСТУП

Стрімкий розвиток цифрових технологій і зростання обсягу даних формують нові вимоги до інфраструктури датацентрів. Сучасні компанії та організації дедалі більше залежать від стабільної роботи інформаційних систем, а отже від надійності та ефективності обчислювальних ресурсів. У цих умовах зростає потреба в технологіях, які дозволяють гнучко розподіляти апаратні потужності, зменшувати експлуатаційні витрати і забезпечувати стабільну роботу сервісів за підвищеного навантаження. Саме тому віртуалізація обчислювальних ресурсів сьогодні розглядається як один із ключових напрямів оптимізації роботи датацентрів.

Актуальність теми зумовлена тим, що традиційні підходи до побудови серверної інфраструктури нерідко призводять до нерівномірного використання фізичного обладнання. У багатьох випадках значна частина потужностей простоює, а масштабування потребує суттєвих фінансових витрат. Віртуалізація дає змогу консолідувати ресурси, забезпечити їх гнучке керування, підвищити рівень безвідмовності та адаптивності інфраструктури. Крім того, вона створює основу для подальшого розвитку хмарних рішень та програмно визначених середовищ.

Метою цієї кваліфікаційної роботи є дослідження технологій віртуалізації обчислювальних ресурсів і визначення шляхів підвищення ефективності роботи датацентрів за їх використання. Передбачається аналіз сучасних рішень, оцінювання їхніх можливостей та розроблення підходів до побудови інфраструктури, яка забезпечує оптимальне використання обладнання.

Отримані результати можуть бути застосовані під час модернізації корпоративних датацентрів, створення серверних ферм, впровадження хмарних сервісів або оптимізації IT інфраструктури підприємств. Вони також можуть слугувати навчальною базою для підготовки спеціалістів у галузі інформаційних технологій.

Об’єктом дослідження є обчислювальна інфраструктура датацентру.

Предметом дослідження є технології і методи віртуалізації, що впливають на ефективність використання ресурсів і загальні показники роботи серверного середовища.

Для досягнення поставленої мети необхідно виконати такі завдання:

- ✓ вивчити теоретичні основи і класифікацію засобів віртуалізації;
- ✓ проаналізувати сучасні платформи, які застосовуються для створення віртуалізованої інфраструктури;
- ✓ оцінити вплив віртуалізації на продуктивність і надійність серверних систем;
- ✓ розробити модель або концепцію оптимізованої інфраструктури датацентру;
- ✓ визначити рекомендації щодо практичного впровадження отриманих результатів.

РОЗДІЛ 1.
АНАЛІЗ СТАНУ ПИТАННЯ В ТЕОРІЇ ТА ПРАКТИЦІ

1.1 Характеристика сучасних дата центрів та їх обчислювальної інфраструктури

Сучасні дата центри є інфраструктурною основою цифрової економіки. Вони забезпечують роботу хмарних сервісів, корпоративних інформаційних систем, електронної комерції, банківських платформ, систем електронного урядування. Фактично будь яка велика організація, яка працює з великими обсягами даних, прямо або опосередковано спирається на ресурси одного чи кількох дата центрів. Тому вимоги до їх продуктивності, доступності, масштабованості та енергоефективності постійно зростають [1-4].

Класичний дата центр включає кілька основних підсистем. По перше, це обчислювальні ресурси, які представлені фізичними серверами загального призначення, блейд серверами та спеціалізованими обчислювальними вузлами для високопродуктивних обчислень. По друге, це системи зберігання даних, які можуть бути організовані у вигляді локально підключених дисків, мережевих сховищ NAS або систем зберігання на основі SAN. По третє, це мережеве обладнання, що забезпечує внутрішню комутацію, доступ до зовнішніх мереж та взаємодію між сервісами. До цього додаються інженерні системи електроживлення, резервування, охолодження, а також системи моніторингу і управління[5].

Історично в багатьох дата центрах використовувалася трирівнева мережева архітектура з виділенням рівня доступу, рівня агрегації та рівня ядра. Така модель добре відповідала потребам, коли основні потоки трафіку проходили між користувачами та серверами. З розвитком віртуалізації та хмарних технологій характер трафіку змінився: істотно зросла частка горизонтальних обмінів між серверами, так званий east west трафік. Це призвело до поширення архітектури leaf spine, у якій кожен комутатор рівня leaf з’єднаний з усіма

комутаторами рівня spine. Такий підхід забезпечує передбачувану кількість переходів між будь якими двома серверами, мінімізує затримки і спрощує масштабування інфраструктури [2-6].

У типовому сучасному центрі обробки даних сервери згруповані в стійки або шаси. До кожної стійки підведено комутатори верхнього рівня (Top of Rack або leaf), які забезпечують підключення серверів до фабрики комутації дата центру. У ролі spine виступають високо продуктивні комутатори, що об’єднують декілька десятків або сотень leaf пристроїв. Така побудова добре узгоджується з віртуалізованими навантаженнями, оскільки дозволяє рівномірно розподіляти віртуальні машини між фізичними хостами та забезпечувати сталу пропускну здатність між ними.

У сучасних публікаціях зазначається, що одним із ключових показників ефективності дата-центру є коефіцієнт використання обчислювальних ресурсів. У традиційних середовищах цей показник часто становив менше половини, оскільки сервери призначалися під вузькоспеціалізовані задачі і не використовувалися повною мірою [1, 3]. Системи моніторингу відстежують рівень завантаження процесорів, обсяг використаної пам’яті, пропускну здатність мережі та параметри енергоспоживання.

Важливим аспектом характеристики дата центру є взаємозв’язок між обчислювальними ресурсами та системами зберігання даних. Для підтримки віртуалізованих середовищ широко застосовуються спеціалізовані сховища SAN, які підключаються до серверів по мережах Fibre Channel або iSCSI. У багатьох сучасних рішеннях використовують гібридні сховища, де поєднано традиційні жорсткі диски та твердотільні накопичувачі. Це дає змогу одночасно забезпечити великий обсяг даних і високу продуктивність під час обробки транзакційних навантажень.

З точки зору експлуатації дата центр характеризується набором показників. До ключових належать рівень доступності сервісів, який часто описують у відсотках «часу безвідмовної роботи», середня затримка в мережі, середній рівень завантаження процесорів, використання оперативної пам’яті та

дискових ресурсів, коефіцієнт енергоефективності PUE. Для традиційних, не віртуалізованих інфраструктур типовою проблемою є низьке середнє завантаження серверів. Часто для гарантування резерву продуктивності та безпеки окремі апаратні платформи відводять під одну важливу систему. Наслідком стає те, що значна частина обчислювальної потужності простоює.

З ростом кількості сервісів та ускладненням мережевих взаємодій управління такою інфраструктурою без засобів централізованого моніторингу та автоматизації стає надто трудомістким. Тому поряд із фізичними пристроями дедалі більшого значення набувають системи, які дозволяють збирати телеметрію, відстежувати навантаження, автоматично балансувати трафік і перерозподіляти ресурси. У цьому контексті віртуалізація виступає фундаментальною технологією, адже саме вона забезпечує гнучкий логічний поділ і консолідацію ресурсів.

Характерною рисою сучасних дата центрів є також інтеграція з хмарною інфраструктурою. Для багатьох організацій поєднуються власні потужності та ресурси публічних хмар. Така гібридна модель дає змогу розміщувати базові критичні сервіси у локальному дата центрі, а пікові навантаження або менш критичні застосунки переносити у публічну хмару. Подібний підхід тісно пов’язаний з концепцією ресурс пулінгу, коли обчислювальні ресурси об’єднуються у спільний пул і динамічно розподіляються між споживачами [4, 7].

Для візуалізації структури сучасного дата центру, в якому поєднано сервери, системи зберігання, мережеву фабрику та віртуалізований рівень, доцільно використати одну з типових схем. Наприклад, у відкритих матеріалах Cisco, присвячених проєктуванню віртуалізованих дата центрів, подаються схеми із двома майданчиками, рівнем ядра, spine і leaf комутаторами, SAN та системами зберігання, як показано на рисунку 1.1.

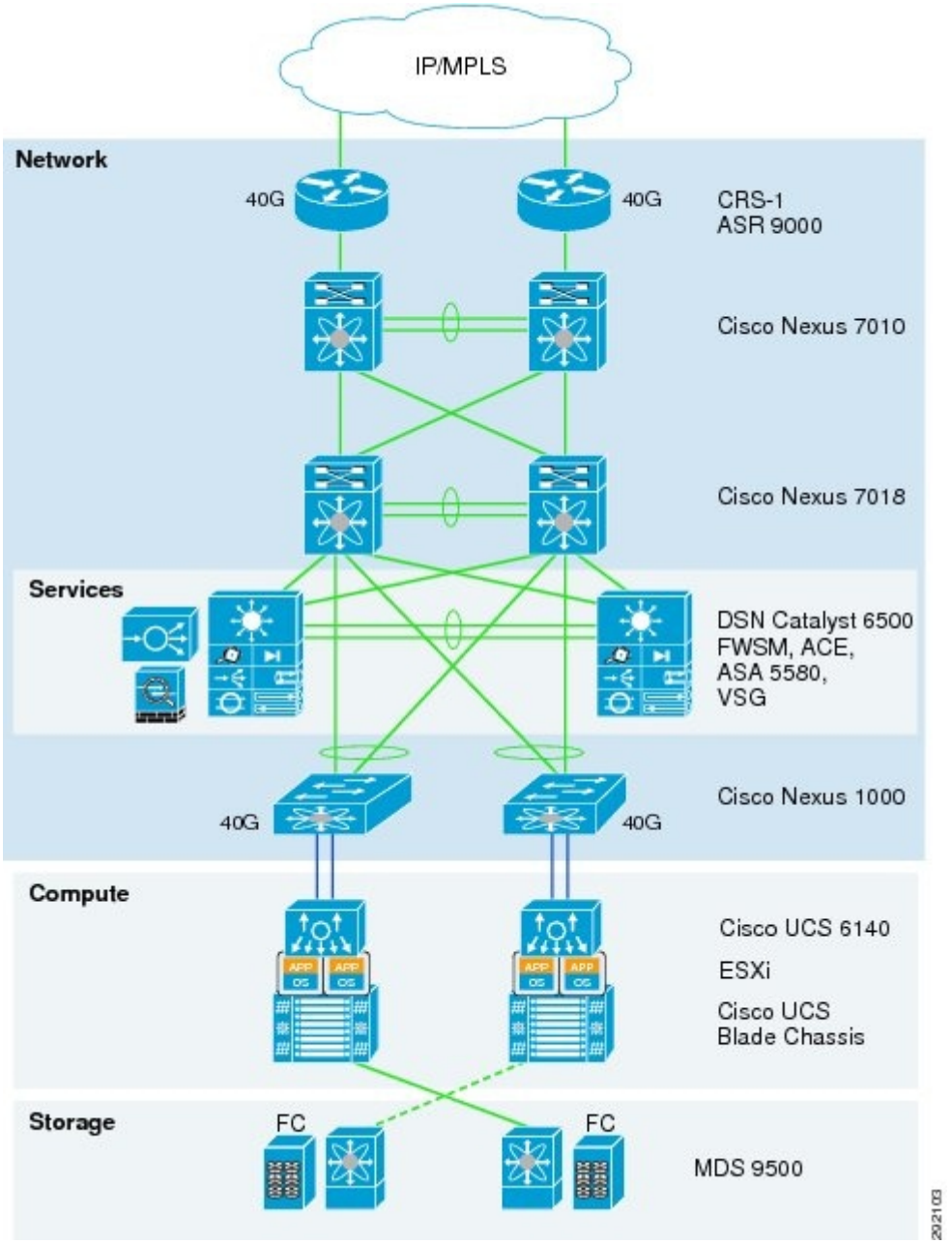

Рисунок 1.1 – Узагальнена архітектура сучасного дата-центру (побудовано за матеріалами Cisco та Juniper) [2, 6].

Загалом характеристика сучасних дата центрів показує, що їхня інфраструктура стала надзвичайно складною. Вона поєднує фізичні ресурси, логічні віртуалізовані рівні, високошвидкісні мережі та інструменти моніторингу. За таких умов саме віртуалізація стає ключовим засобом, який дозволяє привести цю складність до керованого вигляду та створити основу для подальшої оптимізації.

1.2 Теоретичні основи віртуалізації обчислювальних ресурсів

Віртуалізація обчислювальних ресурсів є концепцією, яка передбачає створення абстрактних, логічних представлень фізичних елементів інформаційної системи. До таких елементів належать сервери, дискові системи, мережеві пристрої, робочі станції. Ідея полягає в тому, що користувачі і застосунки працюють не безпосередньо з фізичними пристроями, а з їх віртуальними аналогами, які можуть динамічно створюватися, змінювати конфігурацію, переноситися між хостами та видалятися [8-9].

У багатьох авторитетних джерелах віртуалізацію розглядають як базовий механізм, що забезпечує ресурс пулінг, масштабованість та еластичність хмарних середовищ. Визначення хмарних обчислень, сформульоване NIST, підкреслює, що обчислювальні ресурси провайдера об’єднуються у спільний пул, з якого вони динамічно виділяються споживачам за багатокористувацькою моделлю [10].

Без віртуалізації реалізація такого підходу практично неможлива, оскільки саме вона дозволяє ізолювати навантаження, гнучко розподіляти ресурси та швидко масштабувати середовище.

З історичної точки зору витоки віртуалізації пов’язані з великими мейнфреймами, на яких ще у 60 70 роках минулого століття реалізовувалися віртуальні машини для одночасного виконання декількох операційних систем. Сучасний етап розвитку технології пов’язаний з x86 платформами, на яких виникли популярні гіпервізори та цілі програмні комплекси для побудови віртуалізованих дата центрів. До найбільш відомих рішень належать VMware vSphere, Microsoft Hyper V, KVM, Xen, а також численні комерційні та відкриті платформи, що базуються на цих ядрах.

Ключовою ланкою серверної віртуалізації є гіпервізор. У загальному визначенні гіпервізор це програмне забезпечення, яке дозволяє на одному фізичному сервері запускати кілька віртуальних машин, кожна з яких має власну операційну систему. Гіпервізор бере на себе завдання з керування доступом до

процесорних ресурсів, оперативної пам’яті, пристроїв введення виведення та мережевих інтерфейсів.

Прийнято розрізняти два основні типи гіпервізорів. Гіпервізори першого типу, або bare metal, встановлюються безпосередньо на апаратне забезпечення, замінюючи собою традиційну операційну систему. Вони забезпечують високий рівень продуктивності та ізоляції, тому найчастіше застосовуються в дата центрах. Прикладом є Microsoft Hyper V у серверному варіанті, VMware ESXi та низка інших промислових продуктів.

Гіпервізори другого типу працюють поверх базової операційної системи, використовуючи її ядро для доступу до апаратних ресурсів. Вони зазвичай застосовуються для тестових, лабораторних або робочих станційних сценаріїв, де зручність і сумісність важливіші за максимальну продуктивність [8, 11, 12].

Архітектура гіпервізорної віртуалізації зазвичай зображається у вигляді багатошарового рисунка. Внизу знаходиться апаратний рівень: процесор, оперативна пам’ять, контролери введення виведення, мережеві інтерфейси, системи зберігання. Вище розташований гіпервізор, який відповідає за розподіл ресурсів та забезпечення ізоляції. Над ним працюють віртуальні машини з гостьовими операційними системами та прикладним програмним забезпеченням. У низці джерел додатково виділяють рівень керування, де працюють компоненти, що забезпечують централізований моніторинг та оркестрацію віртуальних машин.

У літературі поширені схеми, які зображають архітектуру гіпервізора: апаратний рівень, шар гіпервізора, гостьові операційні системи та застосунки. Подібна модель демонструє, як завдяки розмежуванню шарів забезпечується ізоляція та незалежне функціонування віртуальних машин [11].

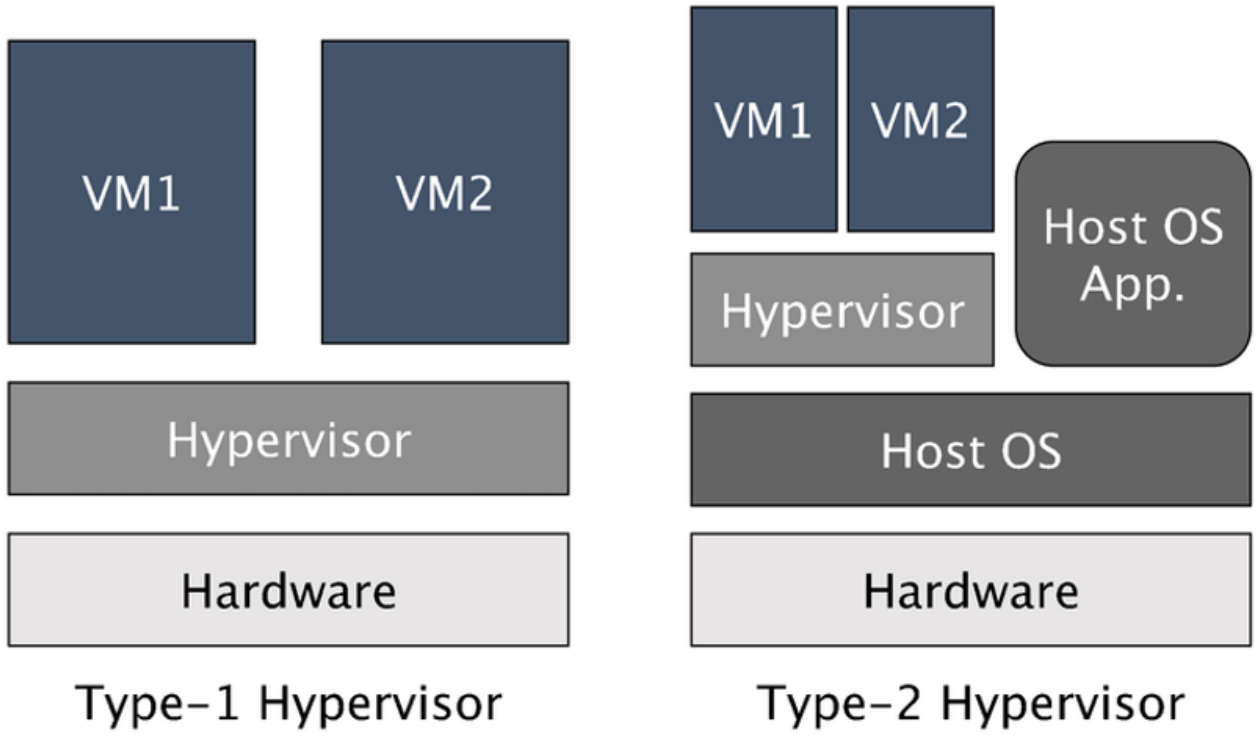

Рисунок 1.2 – Узагальнена архітектура гіпервізорної віртуалізації [8, 11].

Слід зауважити, що віртуалізація не обмежується лише серверним рівнем. Розрізняють також віртуалізацію сховищ, віртуалізацію мереж, віртуалізацію робочих місць, додатків. Віртуалізація сховищ передбачає об'єднання фізичних дисків у логічні томи та пулі, які подаються серверам як абстрактні ресурси. Віртуалізація мереж включає використання віртуальних комутаторів, маршрутизаторів, мережевих функцій, а також технологій програмно визначених мереж [9]. Завдяки цьому на спільній фізичній інфраструктурі можуть співіснувати кілька ізольованих віртуальних мереж, які мають власні політики безпеки та маршрутизації.

Окремий напрям розвитку становить контейнерна віртуалізація, де замість відокремлених віртуальних машин використовуються контейнери, що розділяють ядро операційної системи, але мають ізольований простір процесів і файлову систему [13]. Порівняння гіпервізорної та контейнерної моделей часто наводять у вигляді рисунків, де з одного боку показано стек апаратне

забезпечення, гіпервізор, гостьові ОС та застосунки, а з іншого апаратне забезпечення, базову ОС, контейнерний рушій та контейнери.

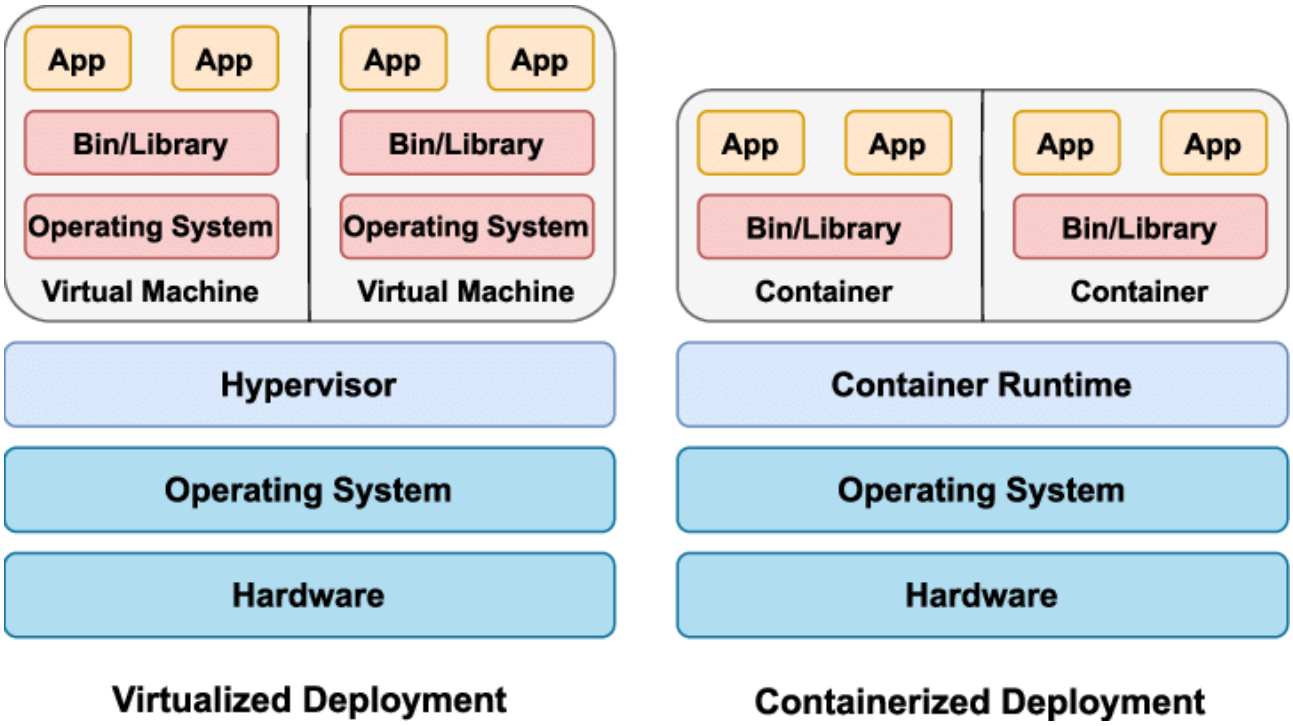

Рисунок 1.3 – Порівняння класичної віртуалізації та контейнеризації (адаптовано за матеріалами Docker та Red Hat) [13].

З теоретичної точки зору віртуалізація тісно пов’язана з поняттями абстракції, мультиплексування та ізоляції. Абстракція означає подання фізичного ресурсу у вигляді стандартного інтерфейсу, незалежного від конкретної реалізації. Мультиплексування полягає у поділі ресурсу між кількома споживачами, коли гіпервізор або інша віртуалізаційна платформа планує час доступу до процесора, розподіляє блоки пам’яті, організовує черги операцій введення виведення. Ізоляція гарантує, що збої або перевантаження в одній віртуальній машині не впливають на роботу інших, а дані залишаються захищеними в рамках визначених політик безпеки.

Для потреб дата центру особливо важливими є два аспекти: можливість динамічного масштабування та централізоване управління. У більшості платформ віртуалізації реалізовані механізми живої міграції віртуальних машин

між хостами, балансування навантаження, автоматичного перезапуску сервісів у разі відмови вузла. Завдяки цьому інфраструктура набуває властивостей еластичності та відмовостійкості, які раніше було складно досягти в суто фізичному середовищі.

Таким чином, теоретичні основи віртуалізації формують базу для розуміння того, як саме віртуалізаційні технології впливають на роботу дата центру, як вони змінюють підхід до планування ресурсів, забезпечення безпеки, резервування і масштабування. Ці положення є необхідною передумовою для подальшого аналізу віртуалізації як інструмента оптимізації.

1.3 Віртуалізація як інструмент оптимізації роботи дата центрів

Використання віртуалізації в дата центрах пов’язане передусім з прагненням підвищити ефективність використання апаратних ресурсів та зменшити експлуатаційні витрати. Багато досліджень та практичних оглядів підкреслюють, що консолідація серверів за допомогою віртуалізації дозволяє скоротити кількість фізичних машин, які необхідні для підтримки заданого обсягу навантаження. Це безпосередньо зменшує витрати на закупівлю обладнання, його обслуговування, електроживлення та охолодження [14–16].

Енергетична складова є одним з найважливіших мотивів впровадження віртуалізації. Емпіричні дослідження показують, що серверна віртуалізація може суттєво знизити споживання електроенергії дата центром за умови правильного планування консолідації навантажень.

Після об’єднання навантажень на меншу кількість серверів деякі фізичні пристрої можна повністю вивести з експлуатації або перевести у енергозберігальний режим. Це не лише зменшує рахунки за електроенергію, а й знижує потребу в системах охолодження та площі, необхідній для розміщення обладнання.

Окрім економії ресурсів, віртуалізація змінює підхід до масштабування інфраструктури. Якщо раніше нарощування потужностей вимагало тривалої процедури закупівлі, доставки, встановлення та налаштування фізичних серверів, то у віртуалізованому середовищі додавання нових віртуальних машин відбувається значно швидше. Адміністратор може автоматизувати процес розгортання стандартних шаблонів систем, а розподіл ресурсів між ними регулюється політиками. Це особливо важливо для організацій з нерівномірним або сезонним навантаженням, де попит на обчислювальні ресурси суттєво змінюється у часі [14, 17].

Суттєвою перевагою віртуалізованих дата центрів є можливість покращити показники доступності та відмовостійкості. Більшість промислових платформ віртуалізації підтримує механізми, які дозволяють переносити віртуальні машини між фізичними хостами без переривання роботи сервісів. Це дає змогу виконувати регламентні роботи, оновлення та модернізацію обладнання з мінімальними простоями. Крім того, у разі відмови окремого сервера система може автоматично перезапустити віртуальні машини на інших хостах у кластері. Такі можливості безпосередньо впливають на рівень доступності сервісів і зменшують ризики простоїв бізнес критичних систем [15].

Віртуалізація спрощує реалізацію резервного копіювання та відновлення після збоїв. Багато рішень дозволяють створювати знімки віртуальних машин, копіювати їх між майданчиками, підтримувати асинхронну реплікацію. У разі аварії можна відносно швидко відновити роботу сервісів, розгорнувши віртуальні машини на резервній інфраструктурі. Це суттєво скорочує показники середнього часу відновлення та знижує втрати від зупинки сервісів [16].

Не менш важливою є оптимізація процесів управління. Віртуалізовані дата центри зазвичай супроводжуються впровадженням централізованих систем моніторингу і оркестрації, які дозволяють адміністратору працювати не з окремими фізичними серверами, а з логічними ресурсами. Наявність єдиної консолі керування значно полегшує контроль за станом інфраструктури, дає змогу автоматизувати рутинні операції, визначати та реалізовувати політики

безпеки, а також швидко реагувати на інциденти. У низці оглядів підкреслюється, що саме спрощення управління є одним із ключових нематеріальних, але дуже відчутних ефектів віртуалізації.

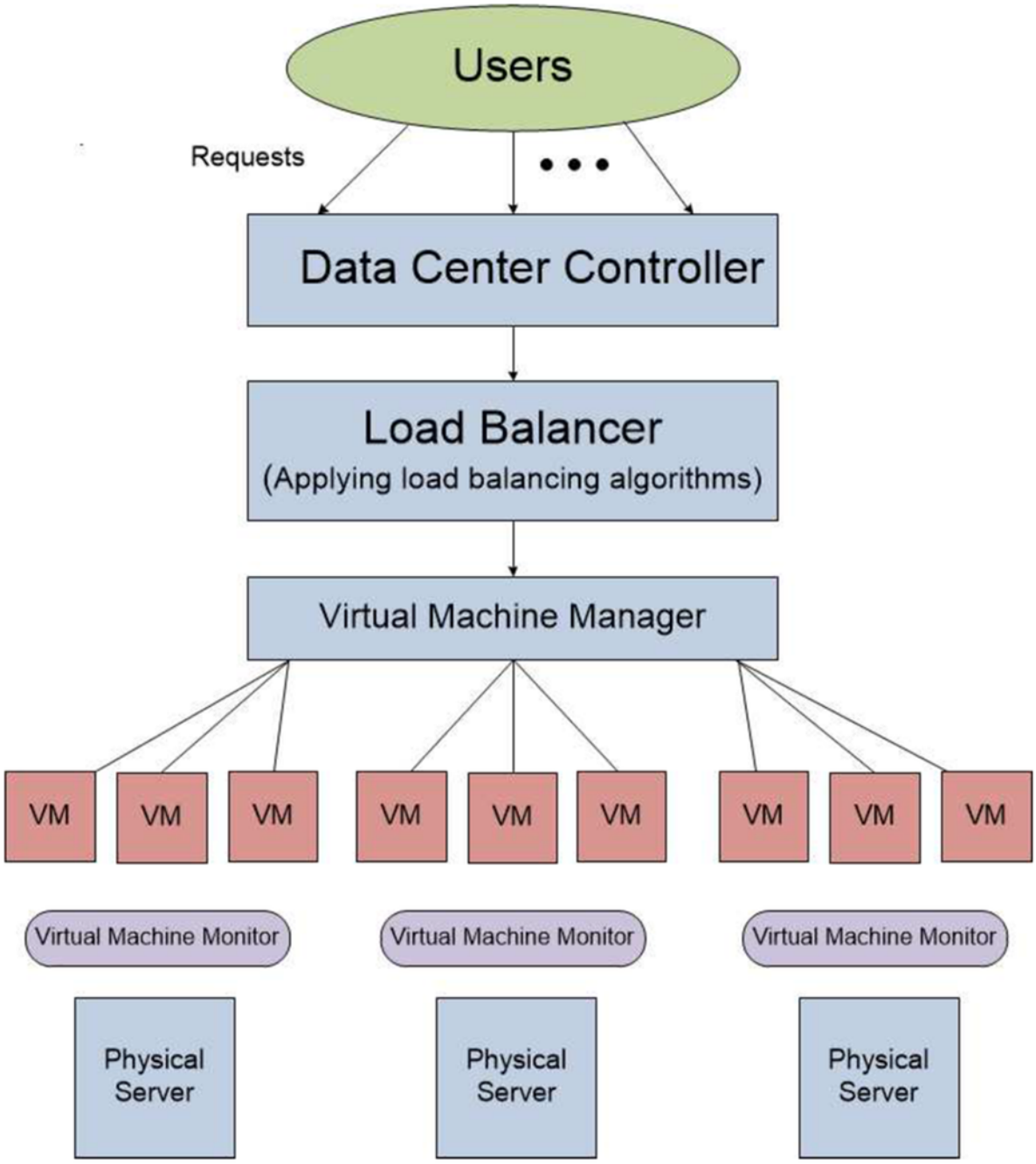

Рисунок 1.4 – Логічна модель віртуалізованого дата-центру з балансуванням навантаження [15, 17].

Разом з тим, впровадження віртуалізації потребує обережного підходу. При неправильному плануванні можлива поява ефекту надмірної консолідації,

коли на одному хості розміщується занадто багато віртуальних машин. Це може призвести до конкуренції за ресурси та деградації продуктивності. Інша проблема пов’язана з «розростанням» кількості віртуальних машин, коли їх створюють без чіткого обліку, що ускладнює управління та підвищує витрати на підтримку. У наукових роботах, присвячених енергоефективності, наголошується на важливості розроблення стратегій розміщення віртуальних машин, балансування навантаження та вимкнення надлишкових ресурсів.

З позиції архітектури оптимізованого дата центру віртуалізація дозволяє поєднати фізичні ресурси в єдине віртуальне середовище, у якому окремі компоненти можуть масштабуватися незалежно. Наприклад, розширення ресурсів для певної групи застосунків може полягати у додаванні нових віртуальних машин до пулу, зміни квот процесорного часу або обсягу оперативної пам’яті, перенесенням навантаження на інший хост.

Ще одним аспектом оптимізації є можливість побудови віртуальних дата центрів поверх фізичної інфраструктури провайдера. У такій моделі клієнт отримує у своє розпорядження логічно ізольоване середовище, яке містить віртуальні мережі, віртуальні машини, сховища та інші об’єкти, але при цьому не має справи з фізичним обладнанням. Провайдер, у свою чергу, може більш ефективно завантажувати свої ресурси, розміщуючи кілька віртуальних дата центрів різних клієнтів на єдиній апаратній базі.

Таким чином, віртуалізація виступає комплексним інструментом оптимізації, який впливає на енергоефективність, вартість володіння, масштабованість, відмовостійкість та керованість дата центрів. Її впровадження змінює сам підхід до проєктування і експлуатації обчислювальної інфраструктури. Подальші розділи роботи будуть присвячені детальнішому аналізу конкретних платформ віртуалізації, методам планування ресурсів, а також розробленню моделі віртуалізованого середовища, орієнтованої на оптимізацію роботи дата центру.

1.4 Постановка завдання

Проведений аналіз сучасних дата центрів, теоретичних основ віртуалізації та її ролі в оптимізації роботи інфраструктури показує, що традиційні фізичні моделі побудови обчислювального середовища вже не відповідають вимогам до масштабованості, гнучкості та ефективності. Встановлено, що зростання обсягу даних, підвищення навантаження на мережеві і серверні ресурси, а також збільшення вимог до доступності створюють суттєві виклики для дата центрів. Визначено, що віртуалізація формує необхідний теоретичний інструментарій для абстрагування фізичних ресурсів та динамічного їх перерозподілу. А також доведено, що застосування віртуалізації забезпечує відчутне підвищення ефективності роботи дата центру за рахунок консолідації навантажень, енергоефективності, покращення показників доступності та можливостей автоматизації.

На основі проведеного аналізу було сформовано проблему дослідження. Вона полягає у необхідності розроблення підходу до впровадження технологій віртуалізації, який дозволить оптимізувати використання ресурсів дата центру, підвищити продуктивність, зменшити витрати на експлуатацію та покращити стійкість до відмов. Окремої уваги потребує виявлення залежностей між типом віртуалізаційної платформи, параметрами навантаження та структурою інфраструктури.

Для досягнення поставленої мети необхідно виконати такі завдання:

- ✓ систематизувати теоретичні відомості про віртуалізацію та архітектуру дата центрів;
- ✓ проаналізувати сучасні платформи віртуалізації та визначити їх переваги й обмеження;
- ✓ дослідити вплив віртуалізації на продуктивність та ефективність використання ресурсів;
- ✓ розробити модель або концепцію віртуалізованої інфраструктури дата центру;

- ✓ оцінити ефективність запропонованого рішення та сформувати рекомендації для практичного впровадження.

РОЗДІЛ 2.

ОБҐРУНТУВАННЯ, ВИБІР ТА РЕАЛІЗАЦІЯ ІНСТРУМЕНТАРІЮ ВІРТУАЛІЗАЦІЇ ДЛЯ ОПТИМІЗАЦІЇ РОБОТИ ДАТА-ЦЕНТРУ

2.1 Вимоги до інструментарію для побудови віртуалізованої інфраструктури

Побудова віртуалізованої інфраструктури починається з ретельного вивчення вимог до інструментарію, який має забезпечити безперебійну роботу дата-центру за умов постійного зростання навантаження. Сучасні центри обробки даних працюють у середовищі, де попит на обчислювальні ресурси змінюється не лише прогнозовано, але й стрибкоподібно. Саме тому інструменти, які використовуються для побудови віртуалізації, повинні забезпечувати одночасно гнучкість, масштабованість, стабільність і енергоефективність [1].

У практиці проєктування дата-центрів першим кроком є визначення характеру робочих навантажень. Якщо планується розгортання великої кількості транзакційних систем, критично важливою стає низька затримка, стабільна робота гіпервізора та здатність виконувати живу міграцію віртуальних машин без зупинки сервісів. Для баз даних важлива підтримка прискорених інструкцій процесора, таких як Intel VT-x або AMD-V, які дозволяють зменшити накладні витрати і передавати частину операцій безпосередньо до апаратних ресурсів.

У випадках, коли дата-центр повинен обслуговувати аналітичні системи чи обробку великих масивів даних, основним пріоритетом стає продуктивність, пропускна здатність мережі та здатність інструментарію працювати з великими дисковими масивами. Важливим є не просто швидке зчитування й запис даних, а забезпечення їх цілісності в умовах високої інтенсивності операцій. Для цього платформа віртуалізації має підтримувати розподілені файлові системи, які

дозволяють оптимізувати роботу зі сховищем і запобігти виникненню «вузьких місць».

Окрему увагу під час вибору інструментарію приділяють підтримці кластерних технологій. Кластер гіпервізорів є основою сучасного дата-центру, оскільки забезпечує стійкість до відмов та можливість масштабування без зупинки сервісів. Здатність платформи виконувати балансування навантаження між хостами, автоматично переміщувати віртуальні машини і перерозподіляти ресурси у відповідь на зміну навантаження визначає рівень її придатності для використання в інтенсивних середовищах.

Кластери також потребують спільної системи зберігання, яка дозволяє віртуальним машинам зберігати дані незалежно від фізичного місця їх розміщення. Розподілені файлові системи, такі як Ceph, забезпечують стійкість до відмов і гарантують автоматичне відновлення у разі втрати вузла. Якщо платформа віртуалізації не має передбаченої інтеграції з подібними технологіями, масштабованість і надійність дата-центру будуть суттєво обмежені [5].

З точки зору мережевої інфраструктури важливою є підтримка програмно-визначених мереж (SDN), які дозволяють керувати маршрутизацією та балансуванням навантаження через програмні механізми. Це спрощує побудову сегментованих мереж, забезпечує гнучке керування політиками безпеки та дозволяє централізовано змінювати мережеві конфігурації. Платформа повинна забезпечувати можливість створювати віртуальні комутатори, маршрутизатори і мережеві функції, що працюють незалежно від фізичної топології.

Для підвищення стабільності дата-центру інструментарій також має надавати можливість інтеграції з системами моніторингу, такими як Prometheus або Zabbix, які дозволяють відстежувати стан віртуальних машин, ресурсів та мережевих компонентів у режимі реального часу. Важливим є не лише збирання інформації, але й аналіз поведінки системи, виявлення аномалій, попередження аварійних ситуацій та автоматичне реагування на зміни.

Оскільки сучасні дата-центри активно переходять до використання контейнерних технологій, інструментарій віртуалізації має підтримувати інтеграцію з Docker, LXC та Kubernetes. Це забезпечує можливість побудови гібридної інфраструктури, у якій на одному фізичному сервері можуть одночасно працювати як віртуальні машини, так і контейнери [7].

Для наочного представлення структури обов'язкових вимог до інструментарію використовується відповідна схема, яка демонструє взаємозв'язок апаратного забезпечення, гіпервізорів, систем керування та контейнерних технологій.

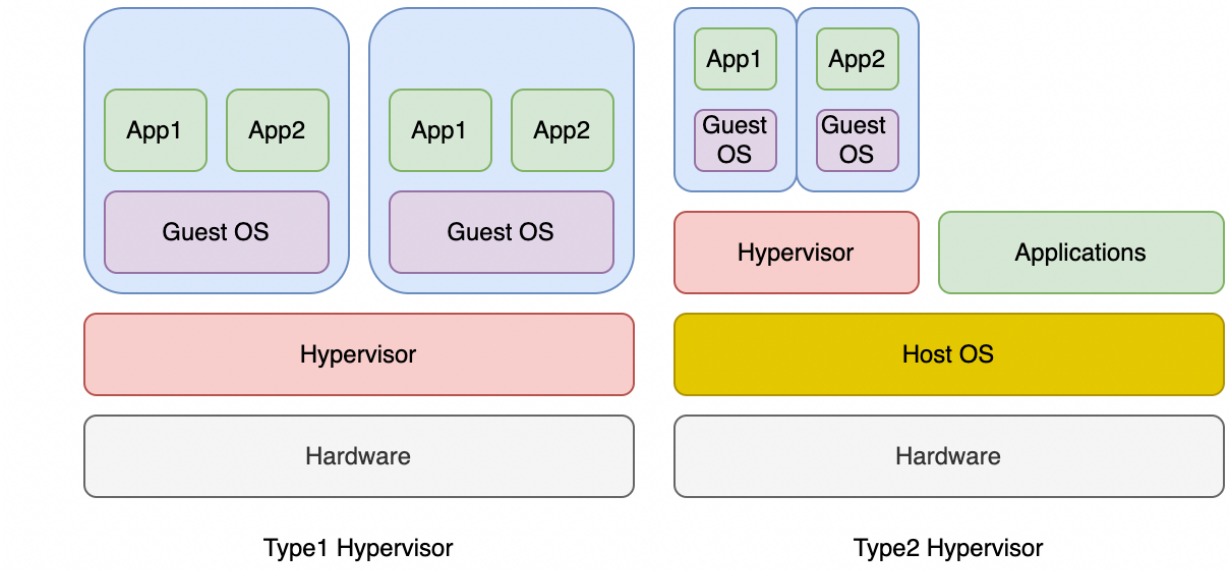

Рисунок 2.1 – Узагальнена структура платформи віртуалізації [1].

Сукупність зазначених вимог дозволяє сформувати критерії, за якими відбуватиметься оцінювання платформ віртуалізації у наступних підрозділах. Правильний вибір інструментарію визначає ефективність роботи дата-центру на роки вперед, тому детальний аналіз є критично важливою частиною проєктування.

На рисунку 2.1 показано багаторівневу структуру сучасної віртуалізації, у якій апаратні ресурси сервера абстрагуються гіпервізором і розподіляються між ізольованими віртуальними машинами. Ця схема наочно показує

взаємозв’язок фізичної інфраструктури, гіпервізора та програмного середовища, підкреслюючи логіку побудови віртуалізованих систем.

2.2 Обґрунтування вибору програмних платформ для побудови віртуалізованого середовища

Вибір програмної платформи для побудови віртуалізованого середовища є одним із ключових етапів формування інфраструктури дата-центру. Від цього рішення залежить не лише поточна продуктивність системи, але й те, наскільки просто вона буде масштабуватися в майбутньому, як швидко реагуватиме на зростання навантаження, наскільки стабільною залишатиметься при збоях та чи зможе інтегруватися з новими технологіями, такими як контейнеризація або програмно-визначені мережі. Платформа віртуалізації має гармонійно поєднувати апаратні можливості дата-центру, вимоги до продуктивності, підтримку кластеризації та готовність до взаємодії з зовнішніми інструментами автоматизації [5].

Початковий етап аналізу передбачає розгляд принципових відмінностей між доступними платформами, серед яких найбільш поширеними є VMware vSphere, Microsoft Hyper-V, Proxmox VE, KVM, Xen та інші продукти корпоративного або відкритого характеру. Кожна з них має свої архітектурні особливості, механізми керування, якість технічної підтримки та набір функцій, що впливають на загальну ефективність роботи.

VMware vSphere тривалий час вважався галузевим стандартом у корпоративному середовищі. Його гіпервізор ESXi створено у вигляді спеціалізованої системи, що працює без сторонньої операційної системи. Такий підхід забезпечує максимальну стабільність, мінімальні накладні витрати та передбачувану продуктивність навіть за умов високих навантажень. Перевагою ESXi є надійний механізм «живої» міграції, вбудовані засоби відновлення після збоїв та добре розвинені інструменти для балансування навантаження між гіпервізорами. У поєднанні з vCenter, що забезпечує централізоване управління

великою кількістю хостів, така платформа прекрасно підходить для великих дата-центрів. Проте суттєвий недолік VMware полягає у високій вартості ліцензій, що обмежує її застосування для невеликих організацій або проєктів з обмеженим бюджетом [6].

Microsoft Hyper-V часто використовують у середовищах, де домінують продукти Microsoft. Гіпервізор інтегровано у Windows Server, що робить його привабливим рішенням для компаній, які вже мають розгорнуту інфраструктуру на базі Active Directory, Windows Server або System Center. Продуктивність Hyper-V у більшості сценаріїв є достатньою, а рівень підтримки конкурентоспроможний. Серед переваг можна назвати низьку вартість впровадження та зручність інтеграції з існуючими корпоративними системами. Проте для побудови масштабної інфраструктури Hyper-V потребує ретельно налаштованого Windows-кластеру, що збільшує обсяг адміністративної роботи.

Окрему нішу займає Proxmox VE, який набув значної популярності завдяки поєднанню відкритого коду, зручного веб-інтерфейсу та здатності працювати як із віртуальними машинами, так і з контейнерами. Proxmox використовує KVM як гіпервізор, що гарантує високу продуктивність та повну сумісність із сучасними процесорами. Завдяки інтеграції з LXC, платформа може одночасно забезпечувати роботу контейнерів. Ця особливість робить її привабливою для дата-центрів, де потрібно поступово переходити до мікросервісної архітектури. Крім того, Proxmox має вбудовані механізми резервного копіювання, реплікації та підтримку розподіленої системи зберігання Ceph, що дозволяє створювати масштабовані й відмовостійкі середовища за порівняно невисокою вартістю [6].

Ще однією платформою, що заслуговує на увагу, є KVM у чистому вигляді. KVM інтегрований у ядро Linux і розглядається не як окремий продукт, а як частина операційної системи. Це дозволяє створити високоадаптивне та тонко налаштоване віртуалізоване середовище. KVM активно використовується у великих хмарних платформах завдяки своїй стабільності та передбачуваній продуктивності. У поєднанні з QEMU та інструментами керування, такими як

libvirt або oVirt, KVM здатний забезпечити рівень гнучкості, який перевищує можливості комерційних гіпервізорів. Проте реалізація подібних рішень вимагає глибоких знань Linux і може бути надто складною для команд з недостатнім досвідом адміністрування.

Для візуального порівняння архітектур різних гіпервізорів наведено схематичне зображення, яке демонструє їх основні структурні відмінності.

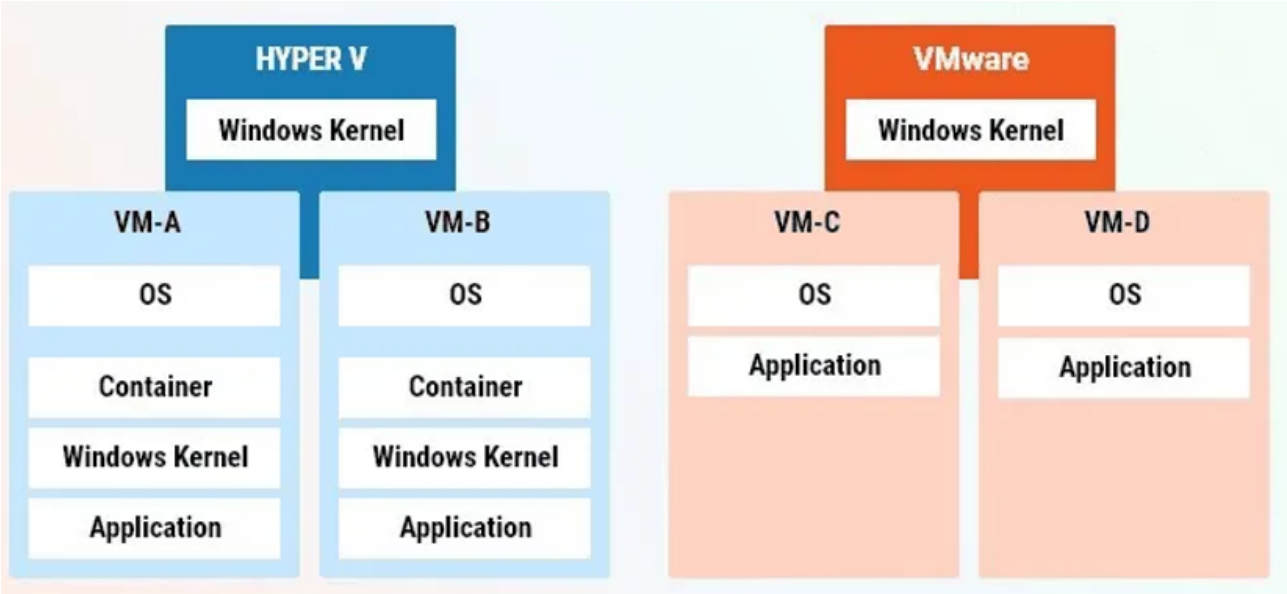

Рисунок 2.2 – Порівняння архітектури VMware ESXi та Hyper-V [5].

Схема наочно показує, що VMware ESXi є мікроядром із мінімальним набором служб, тоді як KVM функціонує як модуль ядра Linux. Hyper-V займає проміжне положення, будучи інтегрованим у Windows Server. Саме ці відмінності визначають принципи роботи, рівень накладних витрат, обсяг доступних функцій та стабільність під час пікових навантажень.

Одним із критеріїв вибору платформи є доступність механізмів автоматизації. У великих дата-центрах без автоматизації неможливо забезпечити стабільне функціонування сотень віртуальних машин. Платформа має підтримувати API, засоби управління через сценарії, можливість використовувати зовнішні оркестратори, такі як Ansible або Terraform. VMware у цьому аспекті пропонує vSphere API, який дозволяє створювати автоматизовані сценарії для керування інфраструктурою. Proxmox VE також має API, що

дозволяє керувати ним через зовнішні системи, а інтегрований інтерфейс спрощує виконання повсякденних завдань.

Важливо враховувати також підтримку високої доступності. У сучасних дата-центрах будь-який простій може спричинити значні фінансові втрати. Тому платформа повинна підтримувати автоматичний перезапуск віртуальних машин у разі збою гіпервізора. Більшість платформ мають механізми, що дозволяють максимально швидко відновлювати роботу системи, однак відмінності між ними полягають у швидкості реакції та складності налаштування.

Ще одним аспектом є підтримка контейнеризації. Дата-центри, які переходять на мікросервісну архітектуру, потребують платформи, що дозволяє одночасно працювати з контейнерами та віртуальними машинами. У цьому сенсі Proxmox VE має перевагу, оскільки дозволяє запускати LXC-контейнери без потреби у додаткових компонентах. У той час Hyper-V та VMware потребують додаткових надбудов, щоб забезпечити інтеграцію з контейнерними системами, такими як Docker чи Kubernetes, тоді як KVM може працювати з контейнерами за допомогою зовнішніх засобів Linux.

Обґрунтування вибору платформи базується також на оцінюванні доступності технічної підтримки та активності спільноти. Для відкритих систем важливо, щоб спільнота розробників була достатньо великою і здатною підтримувати продукт у довгостроковій перспективі. Proxmox відповідає цим вимогам, оскільки має активну спільноту і регулярно оновлюється. Для комерційних платформ важливо, щоб виробник гарантував довгострокові оновлення, стабільні релізи та технічну підтримку. У випадку VMware та Microsoft ці умови виконуються, проте пов'язані з високою вартістю.

Таким чином, вибір платформи для віртуалізованого середовища залежить не від одного параметра, а від сукупності технічних, економічних і стратегічних чинників. Проаналізовані платформи мають свої сфери застосування: VMware ідеально підходить для великих корпорацій, Hyper-V для Microsoft-орієнтованих середовищ, KVM більше пасує для масштабованих хмар, тоді як Proxmox VE пропонує баланс гнучкості, відкритості та широкої

функціональності [5-7]. Саме тому його часто розглядають як оптимальний інструментарій для побудови сучасного, економічного і водночас надійного дата-центру.

2.3 Роль контейнеризації та оркестрації у сучасних дата-центрах

У процесі розвитку сучасних дата-центрів виникла потреба не тільки у віртуалізації апаратних ресурсів, але й у створенні гнучких середовищ для розгортання програмних компонентів, здатних швидко масштабуватися, стабільно працювати під навантаженням і підтримувати безперервність надання сервісів. Традиційна модель віртуальних машин спирається на ізоляцію на рівні операційних систем, що гарантує високу стабільність, проте вимагає значних апаратних ресурсів і створює суттєві накладні витрати. Це стало одним із ключових мотивів для появи контейнеризації, яка пропонує значно легшу, мобільнішу й економнішу модель розгортання застосунків [7].

Контейнери відрізняються від віртуальних машин тим, що не потребують окремих операційних систем. Замість цього вони використовують ядро хостової системи, а ізоляцію забезпечують на рівні процесів і файлових систем. Такий підхід дозволяє помітно зменшити час запуску, скоротити обсяг зайнятої пам’яті та підвищити щільність розгортання. Завдяки цим характеристикам контейнеризація стала центральним елементом мікросервісної архітектури, у якій кожен компонент програмної системи виконує окрему функцію і може масштабуватися незалежно.

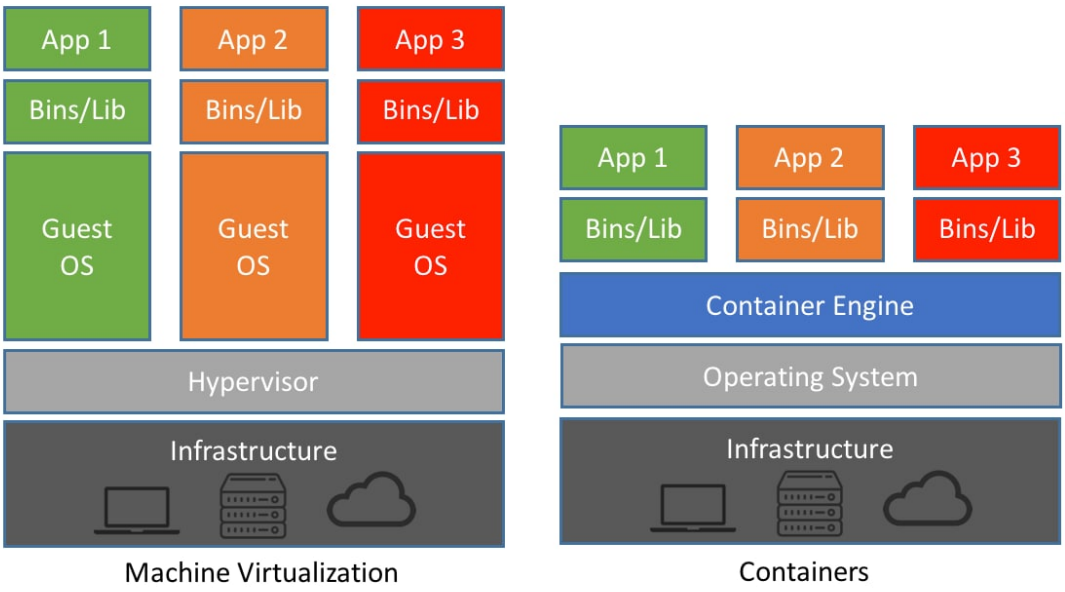

Рисунок 2.3 – Порівняння класичної віртуалізації та контейнеризації [7].

Важливою особливістю контейнеризації є можливість стандартизувати середовище розгортання. Оскільки Docker-образи містять усе необхідне для запуску застосунку, відсутні конфлікти версій, проблему несумісності бібліотек усунуто на рівні архітектури, а розробники можуть бути впевнені, що застосунок працюватиме однаково на будь-якій інфраструктурі. Це значною мірою спрощує як розробку, так і тестування, розгортання та супровід програмного забезпечення. Для порівняння класичної віртуалізації і контейнеризації подано узагальнену схему, яка демонструє їх головні відмінності (рис. 2.3.). Зі схеми чітко видно, що традиційна модель віртуалізації передбачає наявність повноцінної гостьової операційної системи у кожній віртуальній машині. Це забезпечує максимальну ізоляцію, але призводить до значних витрат пам’яті та процесорного часу. У противагу цьому контейнеризація дозволяє запускати багато легких ізольованих середовищ на одному ядрі. Таке рішення особливо ефективне для застосунків, що масштабуються горизонтально, або сервісів із високою інтенсивністю обробки запитів.

Поява контейнерів поставила нове завдання – керування великою кількістю таких модулів у рамках одного середовища. Дата-центри, що працюють із сотнями або тисячами контейнерів, не можуть керувати ними

вручну; для цього потрібен спеціальний інструментарій оркестрації. Найбільш поширеним рішенням став Kubernetes, який фактично став індустріальним стандартом завдяки своїм можливостям автоматичного розподілу навантаження, самовідновлення, масштабування та інтеграції із системами зберігання й мережевими сервісами.

Kubernetes організовує роботу контейнерів у кластери, де логічні компоненти поділені на вузли, поди та контролери. Застосунок може бути розгорнутий у кількох примірниках, а система автоматично слідкує за їхнім станом і перезапускає у разі збою. Завдяки механізмам горизонтального масштабування Kubernetes збільшує або зменшує кількість контейнерів відповідно до змін навантаження. Це особливо важливо для веб-застосунків та сервісів, де кількість запитів може коливатися залежно від часу доби або подій [5].

Однією з головних причин інтеграції контейнеризації з віртуалізацією є бажання побудувати багаторівневу інфраструктуру, де обидві технології доповнюють одна одну. Гіпервізор створює стабільну основу для ізоляції та гарантованого розподілу ресурсів, тоді як контейнери забезпечують швидке розгортання й масштабування прикладних сервісів. У подібній моделі віртуальні машини виступають як серверні вузли для контейнерної платформи, а Kubernetes перетворює їх у гнучкий кластер.

Сучасні дата-центри дедалі частіше будують модель, у якій декілька віртуальних машин виконують роль керованих вузлів Kubernetes, а сам Kubernetes функціонує як автономний шар логічного керування. У таких рішеннях традиційна віртуалізація відповідає за стабільність, а контейнеризація – за ефективність. Цей підхід допомагає зберегти гнучкість під час обробки навантажень, одночасно забезпечуючи високу відмовостійкість системи.

Інтеграція контейнеризації з віртуалізацією також має значення для оптимізації витрат. Віртуальні машини надають можливість виділити визначений обсяг ресурсів для кожного вузла, а контейнерні механізми Kubernetes забезпечують динамічний розподіл контейнерів у межах цих вузлів.

Умовно кажучи, віртуальна машина задає максимальний бюджет ресурсів, а Kubernetes використовує його для розгортання контейнерів до моменту, доки дозволяють виділені ресурси. Це робить систему більш передбачуваною і дозволяє уникати неконтрольованого перевитрачання ресурсів.

Важливим аспектом роботи контейнерів у дата-центрах є їх взаємодія з мережевою інфраструктурою. Kubernetes створює власну внутрішню мережу, у якій кожен контейнер має свою IP-адресу. Це дозволяє забезпечити комунікацію між сервісами, але потребує інтеграції з віртуальними комутаторами, мережевими політиками та механізмами балансування навантаження. Саме тому багато дата-центрів використовують поєднання Kubernetes і програмно-визначених мереж, що дозволяє керувати мережевими потоками на прикладному рівні [5].

Крім того, оркестрація відіграє важливу роль у забезпеченні безпеки. Контейнери можна ізолювати за допомогою політик, контролю доступу та обмеження прав. Kubernetes дозволяє задавати чіткі правила взаємодії між контейнерами, обмежувати мережеві з’єднання і контролювати доступ до конфіденційних даних, які зберігаються у вигляді секретів. Завдяки цьому контейнери стають не просто гнучким, але й безпечним способом розгортання служб.

Сучасні технології оркестрації дозволяють також забезпечувати повну автоматизацію процесів розгортання, оновлення та відновлення. Наприклад, безперервне оновлення (rolling update) дозволяє оновлювати програмні компоненти без зупинки сервісів. У разі збою Kubernetes автоматично повертає стару версію, що мінімізує ризики. Для дата-центру це означає можливість вносити зміни без ризику порушення роботи системи.

Комбінація віртуалізації та контейнеризації створює основу для побудови гнучких хмарних систем. У той час як класична віртуалізація забезпечує стабільну, ізольовану основу для серверної інфраструктури, контейнеризація робить можливим швидке розгортання десятків або сотень сервісів у межах

цього середовища. Оркестрація, у свою чергу, дозволяє контролювати ці сервіси й забезпечувати їхню синхронізовану роботу.

Таким чином, роль контейнеризації та оркестрації у сучасних дата-центрах важко переоцінити. Вони забезпечують гнучкість, швидкість, надійність і економічність, а їх поєднання з віртуалізацією створює комплексну інфраструктуру, здатну ефективно працювати у динамічному середовищі та витримувати різкі зміни навантаження [7].

2.4 Реалізація інструментарію для оптимізації інфраструктури дата-центру

Після проведення аналізу вимог, вивчення можливостей різних платформ віртуалізації та оцінювання ролі контейнеризації в архітектурі сучасних дата-центрів виникає потреба перейти від теоретичних засад до безпосередньої реалізації інструментарію. Саме на цьому етапі відбувається трансформація абстрактних концепцій у конкретну модель інфраструктури, яка здатна забезпечити ефективну, стабільну та масштабовану роботу дата-центру. Реалізація віртуалізованого середовища становить складний багатокомпонентний процес, що охоплює використання апаратних ресурсів, конфігурування гіпервізорів, налаштування мережевої інфраструктури, впровадження механізмів відмовостійкості та адаптацію контейнерних платформ під конкретні задачі.

У сучасних інфраструктурах дедалі більше переваг отримують інтегровані рішення, здатні одночасно підтримувати і віртуальні машини, і контейнери. Це дозволяє гнучко керувати навантаженням, запускати масштабовані сервіси та зберігати можливість ізоляції критично важливих застосунків у межах віртуальних машин. Розглядаючи доступні варіанти, важливо звернути увагу на те, наскільки платформа підтримує різні типи навантажень, як вона реагує на відмови компонентів і чи є достатньо гнучкою для майбутнього розширення.

Одним із найбільш збалансованих рішень є платформа Proxmox VE, яка поєднує гіпервізор KVM та контейнерну технологію LXC. Цей інструментарій має відкриту архітектуру, що дозволяє використовувати його як у невеликих організаціях, так і у масштабованих корпоративних середовищах. На додачу до цього, Proxmox має добре інтегровану систему управління, яка дає змогу адміністратору контролювати весь кластер через веб-інтерфейс без потреби додаткових програмних компонентів. Це підвищує ефективність роботи та зменшує кількість помилок під час конфігурації.

Особливу роль у реальній інфраструктурі відіграє підтримка кластеризації. У випадку Proxmox кластер створюється на основі механізму corosync, який забезпечує зв’язок між вузлами та синхронізацію їхнього стану. Це дозволяє реалізувати високий рівень доступності, коли відмова одного вузла не призводить до зупинки сервісів. Завдяки підтримці механізму live migration віртуальні машини можуть переміщуватися між вузлами без переривання їхньої роботи, що дозволяє виконувати технічне обслуговування або заміну обладнання без зупинок.

Для зберігання даних у кластерній архітектурі зазвичай використовують розподілені файлові системи. Одним із найбільш ефективних рішень є Ceph, який інтегрується з Proxmox на рівні ядра. Ceph забезпечує відмовостійкість, оскільки дані дублюються на кількох вузлах, а система автоматично відновлює репліки у разі втрати окремих дисків або навіть серверів. Такий підхід дозволяє значно знизити ризики пов’язані з апаратними відмовами і забезпечити цілісність даних без необхідності вручну контролювати їх стан [6].

Для пояснення взаємодії вузлів Proxmox, сховища та контейнерної інфраструктури варто використати відповідну архітектурну схему, яку подано на рисунку 2.4.

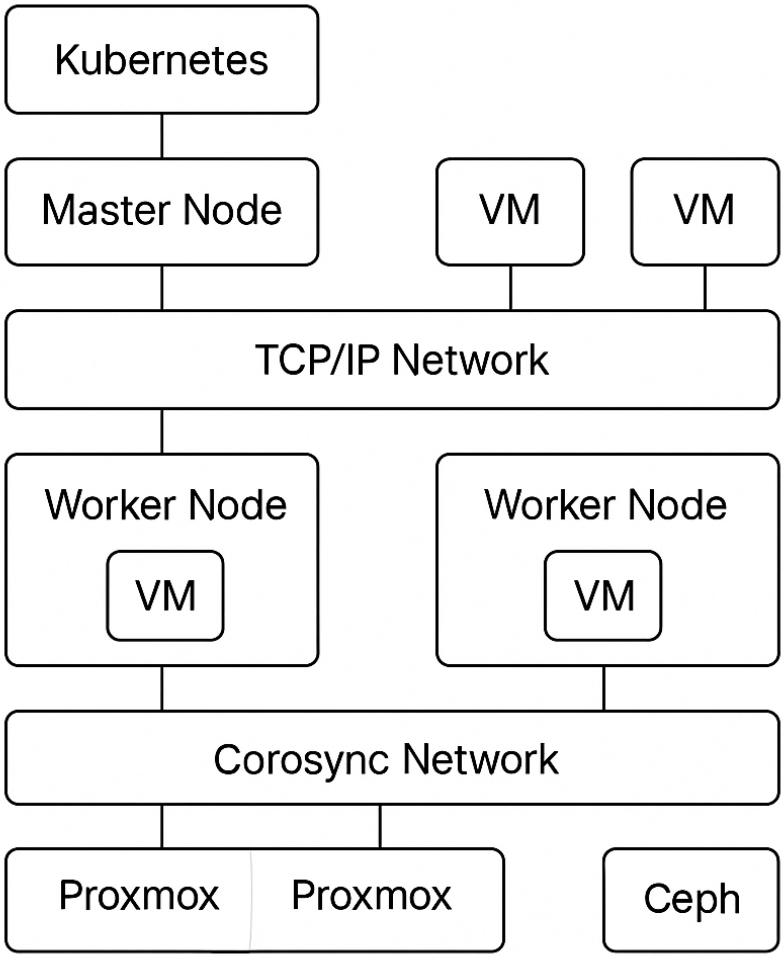

Рисунок 2.4 – Узагальнена архітектура віртуалізованої інфраструктури Proxmox-кластеру.

З рисунка можна побачити, що структура кластеру складається з кількох гіпервізорів, які взаємодіють зі спільним сховищем та логічними мережевими сегментами. Таке розділення дає змогу оптимізувати навантаження, оскільки мережевий трафік, дискові операції та обробка даних виконуються незалежно, не створюючи конфліктів між собою. Ця модель дозволяє передбачувано масштабувати інфраструктуру, додаючи нові вузли до кластеру без потреби переналаштовувати всю систему.

Важливим елементом реалізації інструментарію є мережевий рівень. У віртуалізованих середовищах мережа має балансувати трафік між віртуальними машинами, контейнерами та сервісами управління. Proxmox пропонує підтримку Linux Bridge та Open vSwitch, які дозволяють створювати складні мережеві

топології та сегментувати трафік. Це особливо корисно, коли дата-центр має забезпечувати роботу різних підрозділів або обробку даних з різними рівнями конфіденційності. Мережеві політики можна застосовувати як на рівні віртуальних машин, так і на рівні контейнерів, що підвищує гнучкість інфраструктури [7].

Особливої уваги заслуговує інтеграція контейнеризації з кластером Proxmox. Контейнери LXC забезпечують легке ізольоване середовище, яке підходить для високонавантажених сервісів, що не потребують повноцінних операційних систем. Такі контейнери запускаються швидше, ніж віртуальні машини, і споживають менше ресурсів, що робить їх ідеальними для сервісів з великою кількістю однотипних запитів. Перевага Proxmox полягає в тому, що адміністратор може одночасно керувати і віртуальними машинами, і контейнерами в єдиному інтерфейсі.

Окремим етапом реалізації інструментарію є інтеграція зовнішніх систем оркестрації, таких як Kubernetes. У цьому випадку віртуальні машини Proxmox слугують вузлами для розгортання контейнерного кластеру. Kubernetes забезпечує автоматичний розподіл контейнерів між вузлами, перезапуск сервісів після збоїв, масштабування застосунків відповідно до навантаження та зберігання конфігурацій у централізованому вигляді. Таким чином, на базі віртуалізації формується багаторівнева модель, де традиційні сервери, віртуальні машини та контейнери створюють цілісну систему, здатну реагувати на будь-які зміни у стані інфраструктури.

Реалізація інструментарію також передбачає налаштування системи моніторингу. У кластерах Proxmox можна використовувати вбудовані інструменти відстеження ресурсів. Це дозволяє відстежувати метрики роботи вузлів, дискових систем, контейнерів та віртуальних машин. Важливо, щоб система моніторингу могла не лише збирати дані, але й надсилати повідомлення про аномалії, що дозволяє оперативно реагувати на можливі проблеми.

Під час реальної експлуатації особливої уваги потребують механізми резервного копіювання. Proxmox пропонує вбудований інструмент vzdump, який

дозволяє створювати знімки віртуальних машин та контейнерів. Це робить можливим швидке відновлення у разі помилок або пошкодження даних. Для забезпечення максимального рівня безпеки рекомендується зберігати резервні копії на окремому сховищі, фізично відокремленому від основного кластеру. Це знижує ризик втрати даних у випадку серйозних аварій.

Загалом реалізація інструментарію віртуалізації на базі Proxmox у поєднанні з контейнеризацією та Kubernetes створює гнучку, відмовостійку і масштабовану інфраструктуру. Завдяки поєднанню віртуальних машин і контейнерів дата-центр отримує можливість оптимально розподіляти ресурси залежно від типу навантаження, що дозволяє досягти ефективної роботи навіть у середовищах із різкими коливаннями трафіку та складними вимогами до безпеки [6-7].

РОЗДІЛ 3.

РЕЗУЛЬТАТИ ВИРІШЕННЯ ЗАДАЧІ ОПТИМІЗАЦІЇ РОБОТИ ДАТА-ЦЕНТРУ

3.1 Результати побудови кластера Proxmox VE та перевірка його працездатності

У межах вирішення задачі оптимізації роботи дата-центру було отримано практичний результат у вигляді розгорнутого кластера Proxmox VE, що забезпечує централізоване керування ресурсами, основу для відмовостійкості та можливість подальшої консолідації обчислювальних навантажень. Кластеризація в цьому контексті розглядається як технічний механізм, який перетворює набір фізичних серверів на єдине кероване середовище, де конфігурації, облікові записи, політики та стан інфраструктури синхронізуються між вузлами. Практичний експеримент виконано за сценарієм, описаним у матеріалі Selectel на Habr, з відтворенням логіки дій у вебінтерфейсі Proxmox VE та фіксацією проміжних станів системи.

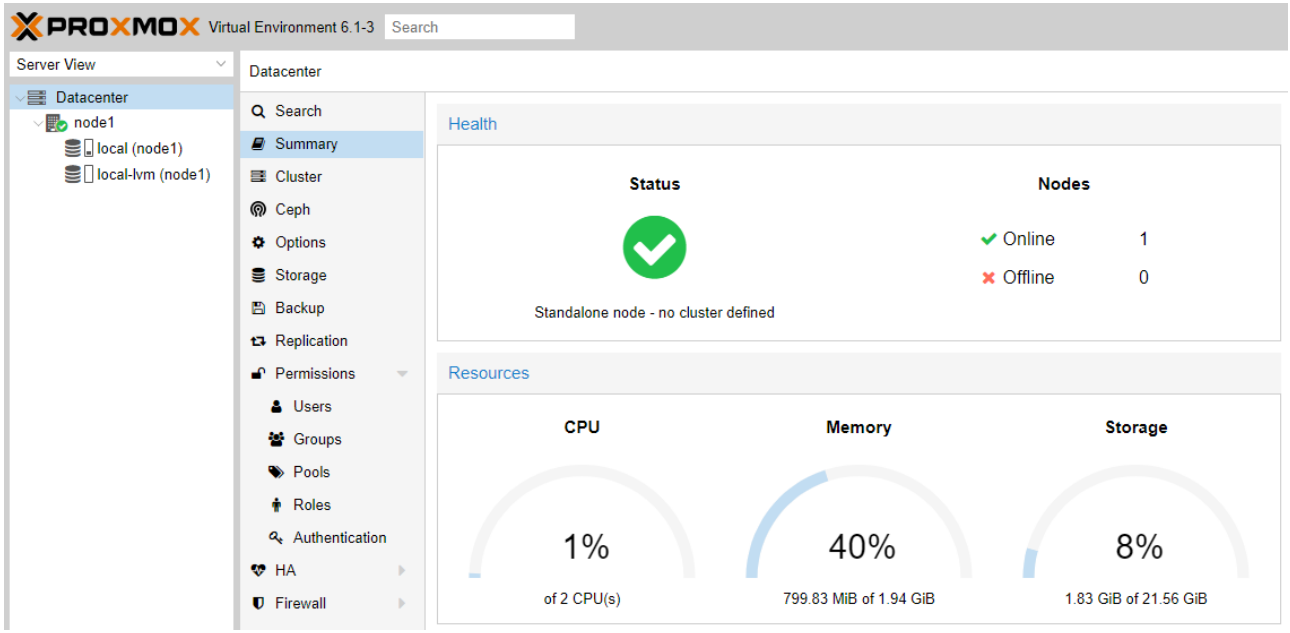

Рисунок 3.1 - Proxmox VE у режимі одиночної ноди (Standalone mode).

Початковий стан інфраструктури відповідав типовій ситуації, коли кожен вузол працює автономно, а керування здійснюється окремо на кожному сервері. У режимі одиночної ноди Proxmox VE демонструє коректну роботу гіпервізора та локальних сховищ, однак відсутні кластерні можливості, зокрема централізований огляд вузлів, спільні політики на рівні Datacenter, а також механізми кворуму та основа для HA. Це добре видно в інтерфейсі, де система відображає стан “Standalone node” і не показує інших вузлів у межах Datacenter.

Перед створенням кластера узгоджуються базові технічні умови, і саме вони істотно впливають на кінцевий результат. По суті, кластер Proxmox тримається на надійній мережевій взаємодії вузлів та на коректній синхронізації служб, які відповідають за обмін повідомленнями і реплікацію конфігурацій. У практичній частині були враховані вимоги щодо низької затримки між вузлами, однакових версій Proxmox VE на всіх серверах, а також відкритих мережевих портів, необхідних для corosync та служб доступу. В офіційній документації Proxmox окремо вказано роль corosync у кластері та використання інструмента pvecm як штатного механізму створення і керування кластером [10]. Також у довідкових матеріалах Proxmox зазначено, що для corosync у кластері зазвичай застосовуються UDP-порти 5404 і 5405, а для адміністративних операцій часто потрібен доступ по SSH, що важливо як на етапі створення, так і під час приєднання вузлів [11].

Створення кластера виконувалося у вебінтерфейсі через розділ Datacenter, вкладка Cluster. На цьому етапі важливо, що операція ініціалізації кластера завжди виконується на першій ноді, яка стає первинною з точки зору формування початкової конфігурації corosync та запуску кластерних служб. Інтерфейс Proxmox вказує на доступність дії “Create Cluster”, що фактично відповідає запуску процедури створення кластерного домену з заданою назвою та мережевим параметром Link 0, який визначає адресацію для кластерного трафіку.

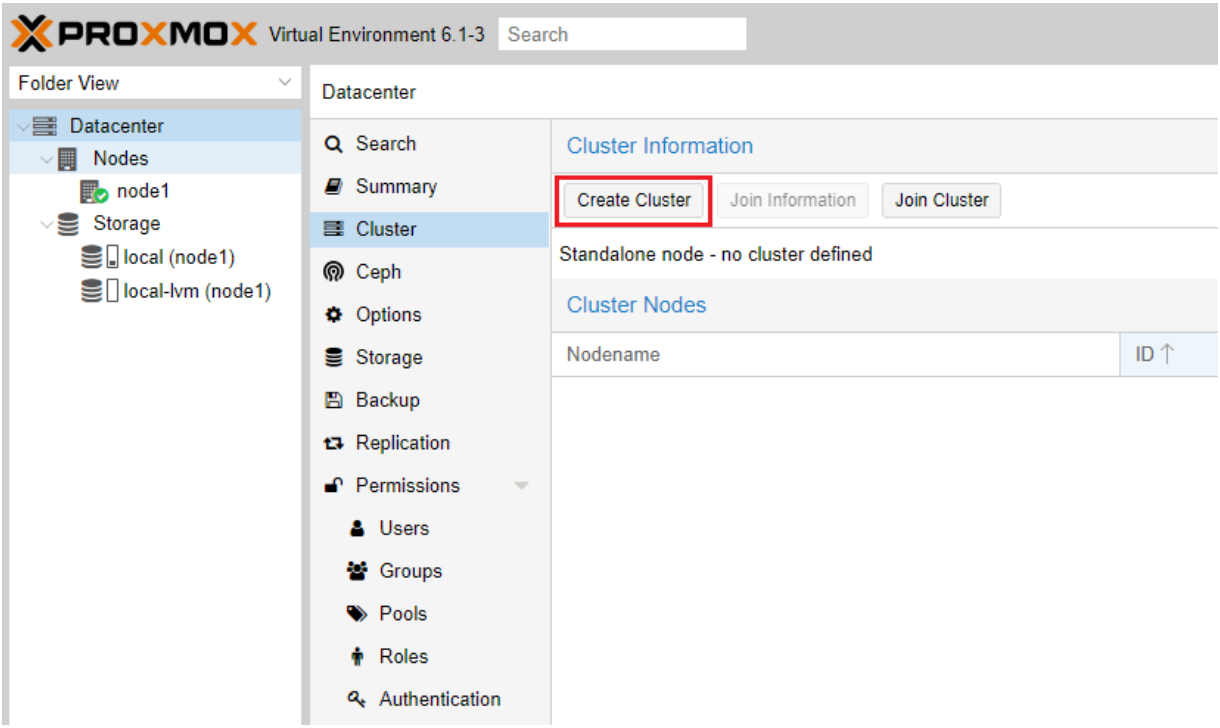

Рисунок 3.2 - Розділ Cluster у Datacenter з кнопкою Create Cluster.

Ключовим параметром створення кластера є вибір мережевого інтерфейсу для Link 0. У практиці це означає вибір IP-адреси, по якій вузли будуть обмінюватися службовими повідомленнями corosync. Саме цей трафік визначає стабільність кворуму, коректність синхронізації конфігурацій та здатність кластера переживати окремі збої. На рисунку показано приклад, де задається ім’я кластера і вибирається адреса мережі для Link 0. У реальній експлуатації доцільно розглядати окрему кластерну мережу, аби службовий трафік не конкурував із трафіком віртуальних машин, але в демонстраційному сценарії достатньо однієї локальної підмережі за умови мінімальної затримки.

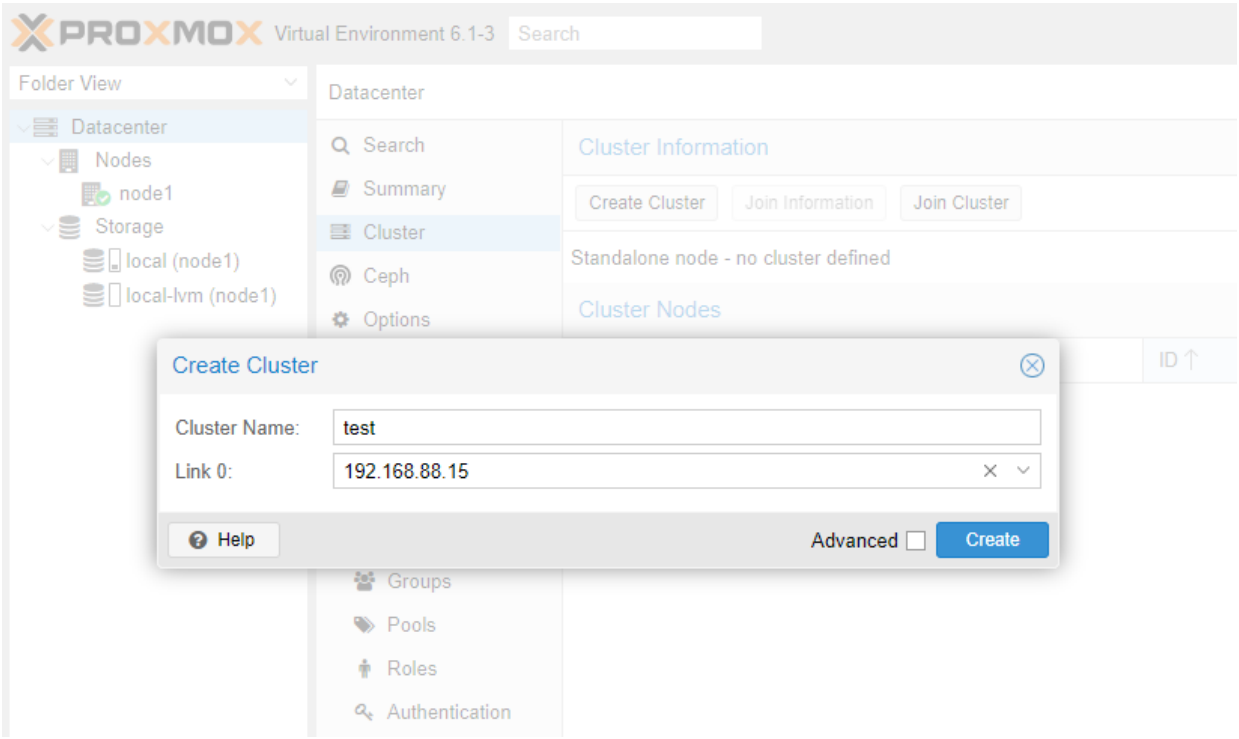

Рисунок 3.3 - Вікно Create Cluster: задання назви кластера та вибір Link 0 (кластерної адреси).

Після підтвердження створення кластера система виконує ряд дій, які важливі для пояснення отриманого результату. По-перше, генерується криптографічний ключ для аутентифікації взаємодії вузлів, який надалі використовується corosync. По-друге, формується конфігурація кластера і записується у файли, які потім реплікуються між вузлами. По-третє, перезапускаються кластерні служби, що переводить ноду в кластерний режим. У вікні завдання Proxmox відображає журнал виконання і маркер успішності “TASK OK”, що інтерпретується як коректне завершення ініціалізації. На рівні офіційної документації це узгоджується з тим, що pvecm виконує створення кластера, спираючись на corosync як на механізм групової комунікації [10].

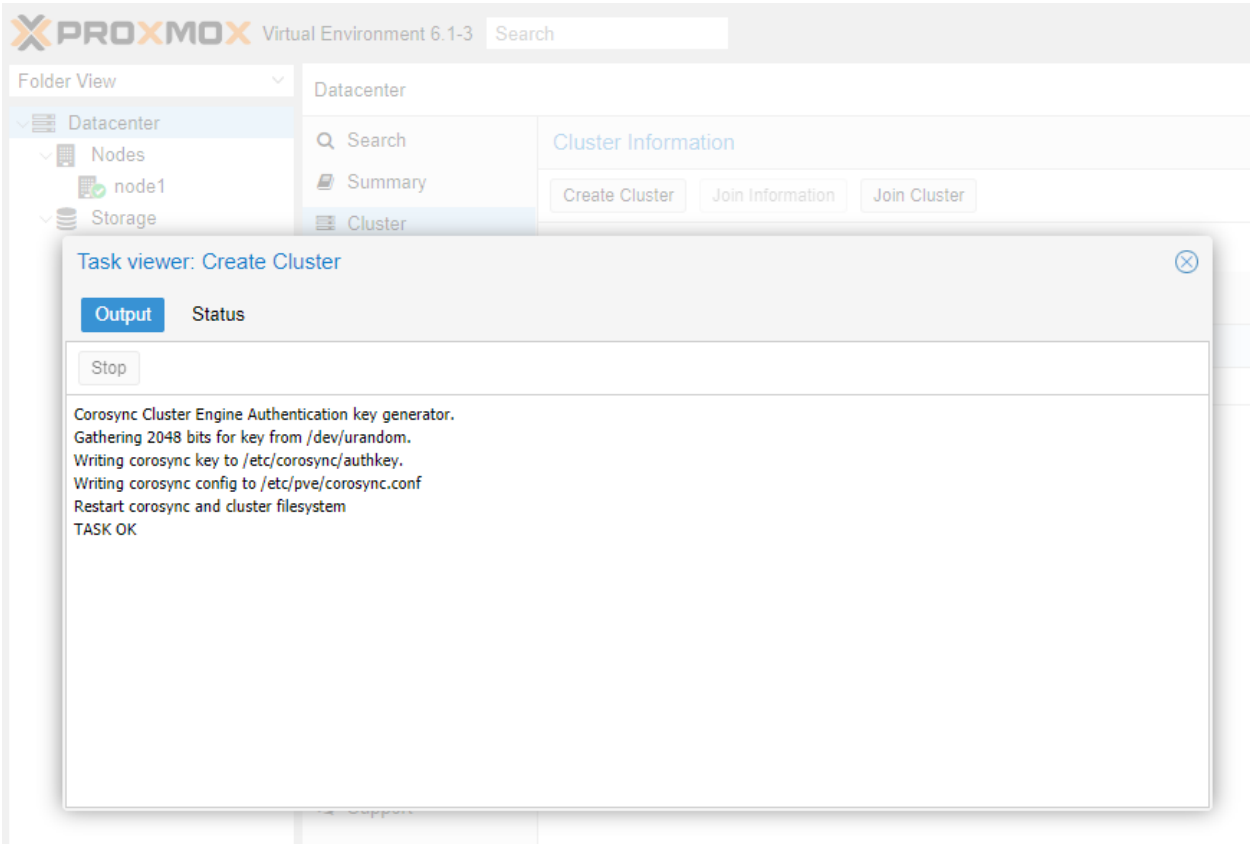

Рисунок 3.4 - Результат операції Create Cluster: журнал дій та статус TASK OK.

Далі фіксується результат переходу вузла у кластерний режим. Система відображає, що кластер має задану назву, а також показує стан кворуму. На ранній стадії, коли кластер складається лише з однієї ноди, кворум можливий, оскільки загальна кількість голосів дорівнює одному, і вузол присутній у системі. В інтерфейсі це видно через позначку “Quorate: Yes”. Практична цінність цього кроку полягає в тому, що адмін отримує підтвердження: кластерні служби активні, конфігурація застосована, а вузол правильним чином зареєстрований у межах кластера.

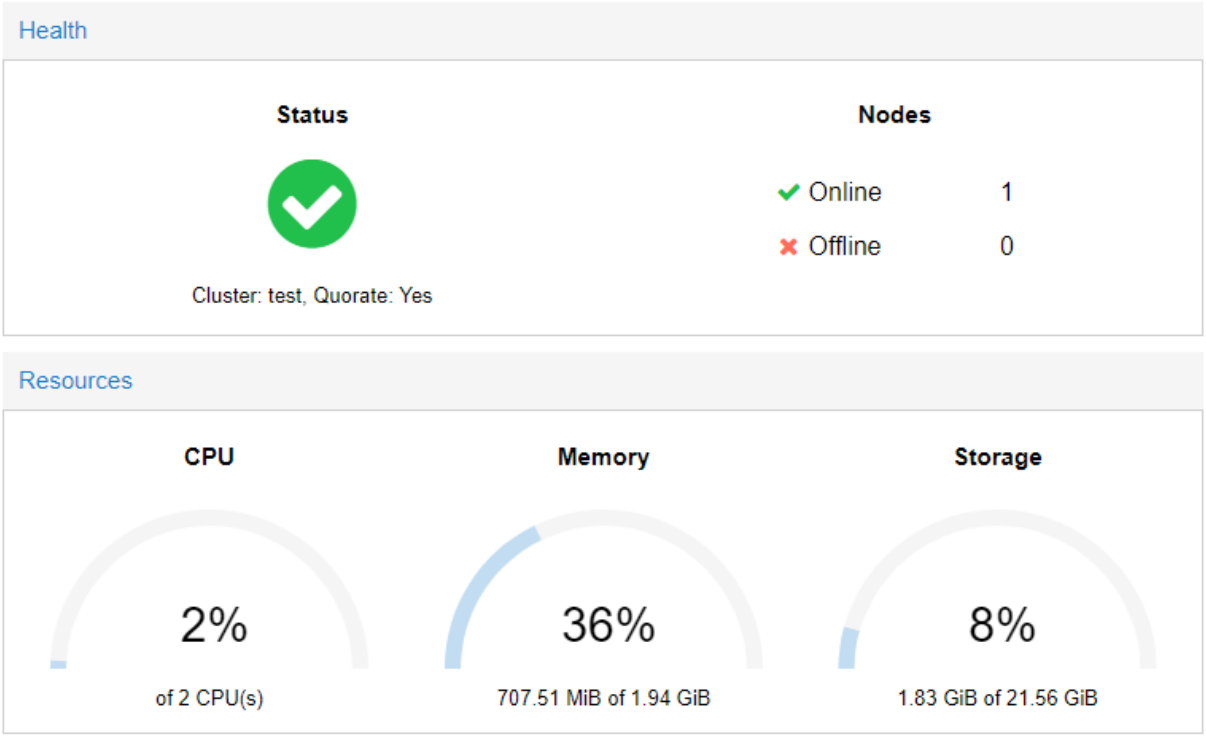

Рисунок 3.5 - Підтвердження створення кластера: відображення Cluster та Quorate: Yes.

Наступним результативним етапом є приєднання додаткових вузлів. У Proxmox VE цей процес може виконуватись через CLI або через вебінтерфейс, але в межах відтворюваного сценарію використано саме GUI, оскільки він наочно демонструє обмін параметрами приєднання. На первинній ноді відкривається розділ Cluster і обирається дія “Join Information”. Фактично система формує пакет параметрів, який містить адресу вузла, цифровий відбиток сертифіката або ключа, а також закодоване поле Join Information. Це зручно тим, що мінімізує ризик помилок під час ручного введення даних на стороні вузла, який приєднується, і одночасно забезпечує перевірку справжності кінцевої точки через fingerprint.

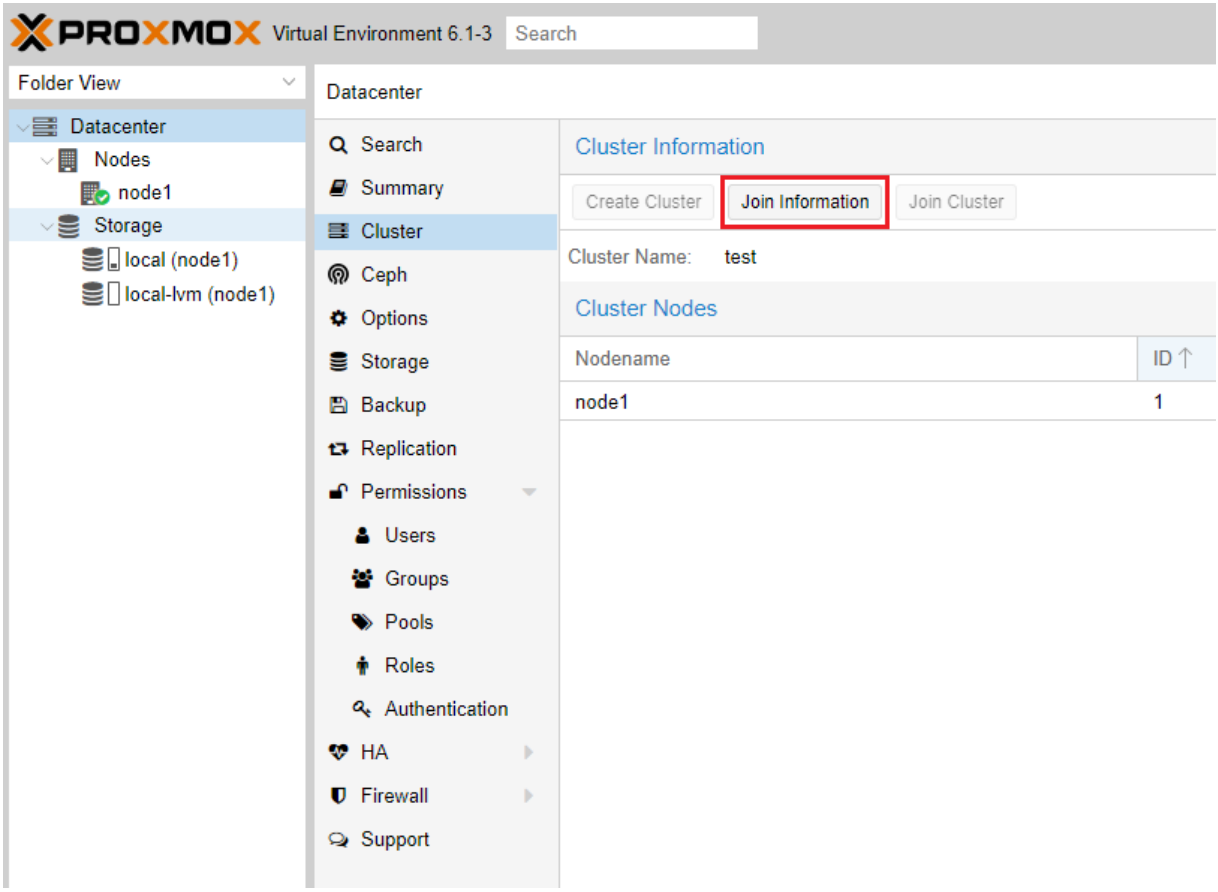

Рисунок 3.6 - Отримання параметрів приєднання: кнопка Join Information на первинній ноді.

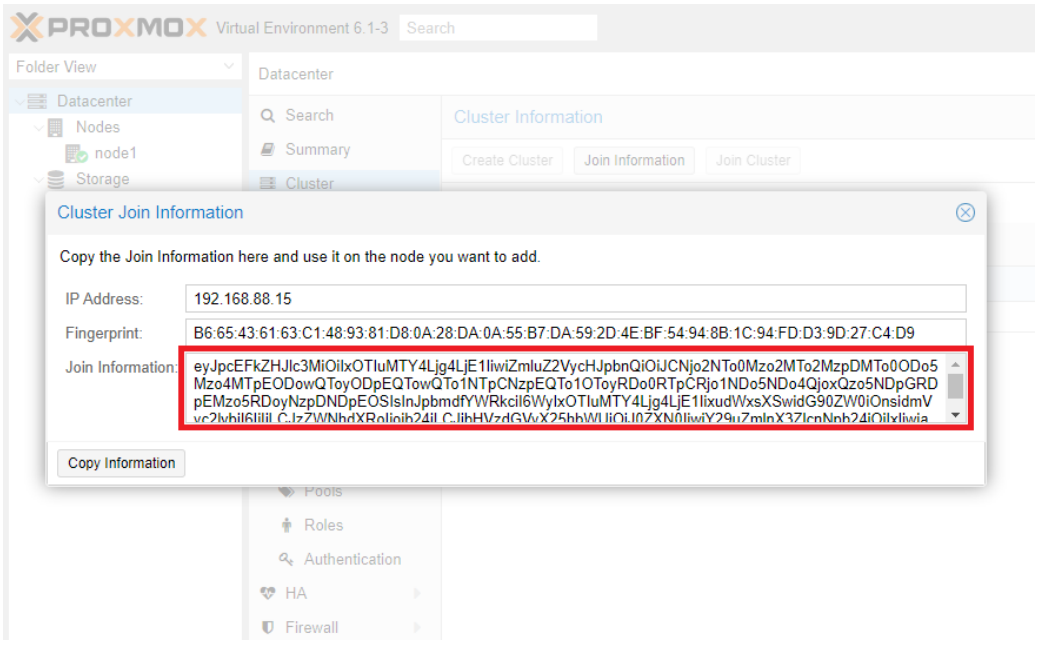

Рисунок 3.7 - Вікно Join Information із закодованими параметрами підключення і fingerprint.

На вузлі, який приєднується, у розділі Cluster використовується дія “Join Cluster”. У вікні приєднання вставляються отримані закодовані параметри, після чого поля Peer Address і Fingerprint підтягуються автоматично. Далі вводиться пароль адміністратора первинної ноди, вибирається мережевий інтерфейс для Link 0 та запускається процедура приєднання. Приклад цього вікна показує, що механізм приєднання поєднує у собі автоматичне заповнення критичних полів і ручне підтвердження доступу, що підвищує керованість процесу у порівнянні з повністю “сліпим” додаванням вузла. З технічної точки зору, на цьому кроці виникає синхронізація кластерної конфігурації, налаштування corosync на новому вузлі та підключення його до спільної моделі стану кластера. Документація Proxmox підтверджує, що pvecm відповідає саме за створення кластера і приєднання вузлів, а групова комунікація реалізується через corosync [10].

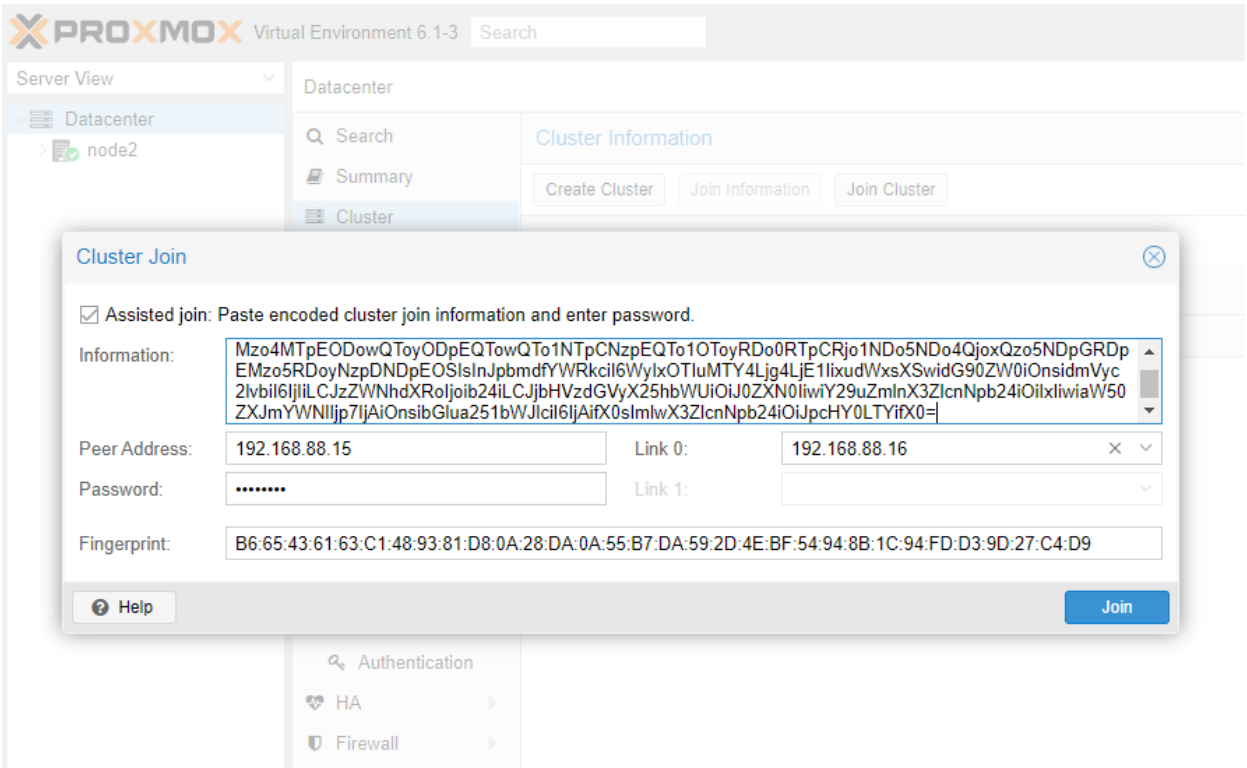

Рисунок 3.8 - Вікно Join Cluster на другій ноді з автоматично заповненими Peer Address та Fingerprint.

У практичному експерименті вузли були додані послідовно, після чого кластер набув повноцінного вигляду з трьома нодами. Результат верифікується в GUI через таблицю вузлів, де для кожного сервера показано ідентифікатор, стан Online, адресу, поточні метрики CPU і пам’яті, а також uptime. Важливо, що після завершення приєднання керування вузлами стає централізованим: адмін бачить усі сервери в одному дереві Datacenter, отримує зведену інформацію про доступні ресурси, може виконувати операції на рівні кластера (наприклад, створення сховищ для всіх нод або налаштування політик), а також переходити до конфігурацій, які без кластеру або складні, або потребують дублювання ручних дій.

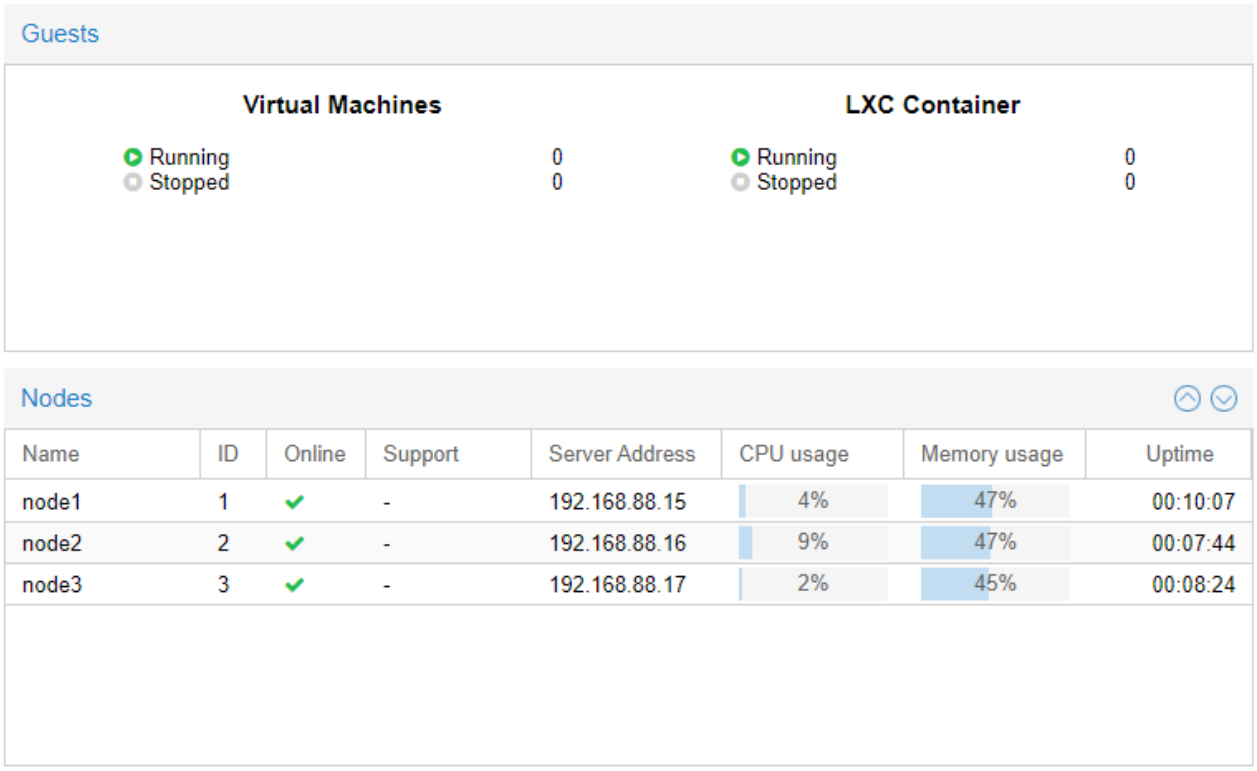

Рисунок 3.9 - Результат кластеризації: перелік нод кластера у Datacenter (node1-node3).

Окремо слід підкреслити отриманий практичний ефект від кластеризації як елемента оптимізації. По-перше, з’являється єдина точка керування, що прямо зменшує адміністративні витрати, оскільки рутинні операції виконуються один раз у масштабі кластера, а не повторюються на кожному сервері. По-друге,

кластер створює фундамент для подальших механізмів, які критичні для оптимізації дата-центру: це балансування і міграції, організація високої доступності, реплікації та більш ефективний розподіл навантаження за рахунок консолідації віртуальних машин на меншій кількості вузлів у моменти низького попиту на ресурси. По-третє, співставлення умов кластерної мережі з довідковими вимогами Proxmox дозволяє зробити висновок, що правильне налаштування мережевої взаємодії, зокрема доступності потрібних портів, є обов'язковою передумовою стабільного кворуму і, відповідно, стабільності всього кластера [11].

3.2 Результати підключення спільного сховища та налаштування High Availability у кластері Proxmox VE

Після успішного формування кластера ключовим практичним кроком стало отримання функціонального результату, який безпосередньо впливає на надійність роботи віртуалізованої інфраструктури. Йдеться про підключення спільного файлового сховища для всіх вузлів і подальше увімкнення механізмів High Availability для вибраних сервісів. Логіка цього рішення полягає в тому, що кластер як об'єднання вузлів дає централізоване керування, але відмовостійкість на рівні віртуальних машин або контейнерів потребує спільного доступу до їхніх дискових даних. Саме тому в практичному сценарії використано мережеве сховище NFS, доступне всім нодам, після чого активовано контроль ресурсу через HA-підсистему Proxmox VE.

Результат підключення спільного сховища оцінювався за двома критеріями. Перший критерій полягає у коректному додаванні NFS як storage на рівні Datacenter, що означає автоматичну доступність цього сховища для кожної ноди кластера. Другий критерій полягає у стабільному відображенні підключеного storage в інтерфейсі одразу на всіх вузлах, що практично підтверджує можливість зберігати диски ВМ і контейнерів у спільному просторі даних. У межах реалізації було повторено типову процедуру Proxmox: у

вебінтерфейсі обрано шлях Datacenter, Storage, Add, NFS, після чого задано ідентифікатор сховища, адресу NFS-сервера та експортовану директорію, яку система підтягує з переліку доступних експортувань.

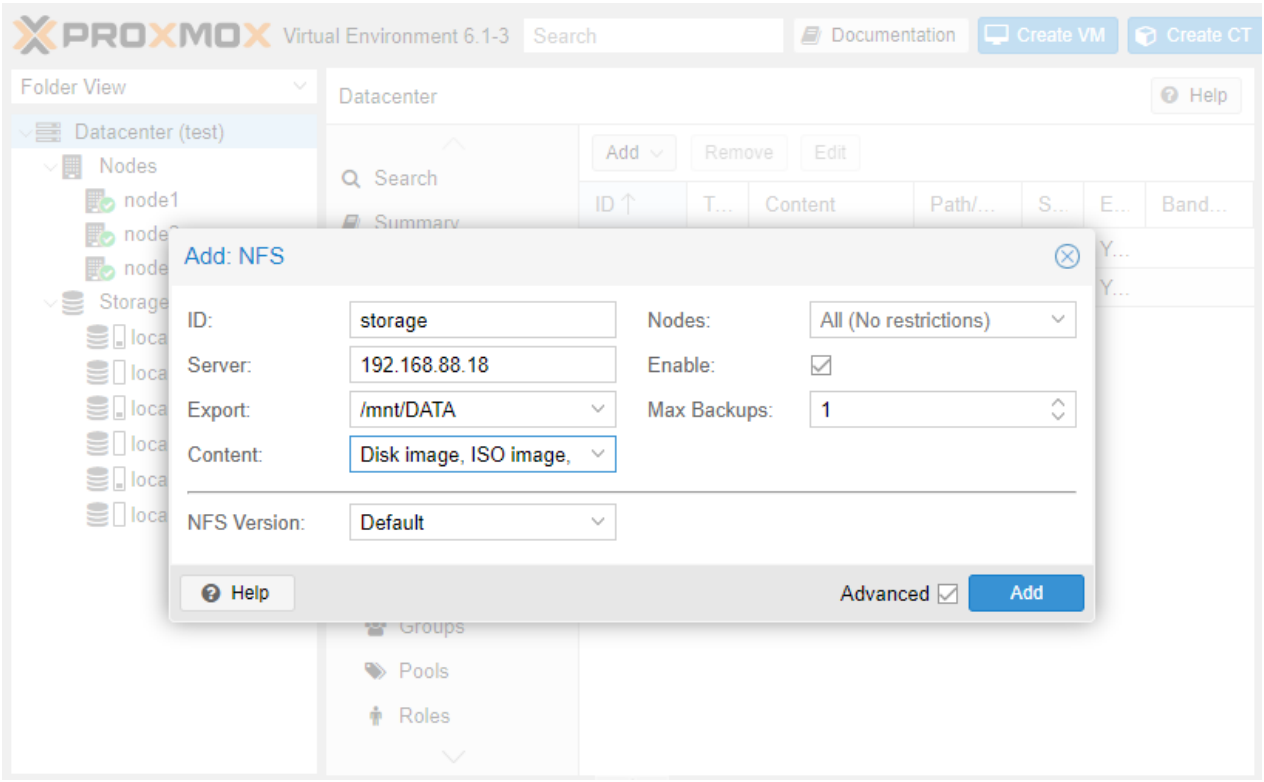

Рисунок 3.10 - Додавання спільного NFS-сховища у Proxmox VE (Datacenter, Storage, Add, NFS).

Заповнення параметрів NFS є не просто формальністю, а способом зафіксувати роль сховища у загальній архітектурі. Ідентифікатор storage використовується як узгоджене ім’я в межах кластера, адреса сервера визначає точку доступу до даних, а параметр Export, який вибирається зі списку, фактично задає кореневий каталог, де будуть розміщуватися образи дисків, резервні копії або темплейти. Вибір типів контенту дозволяє обмежити використання storage, наприклад залишити його лише для дисків віртуальних машин та контейнерів або додатково дозволити зберігання ISO. Після підтвердження операції Proxmox підключає NFS для всіх вузлів кластера, що є важливим практичним спрощенням, оскільки адміністратору не потрібно вручну повторювати налаштування на кожній ноді.

Коректність і завершеність підключення підтверджується візуально у дереві Datacenter, де одна й та сама одиниця storage з'являється під кожною нодою. Це є важливим результатом, тому що саме спільне сховище виконує роль "точки істини" для даних ВМ і контейнерів. Якщо відмова станеться на рівні вузла, дані не залишаються прив'язаними до локальних дисків сервера, а залишаються доступними для іншої ноди, яка може перезапустити ресурс із тим самим дисковим станом. Така властивість і є базовою передумовою для працездатного HA в Proxmox VE, що прямо зазначається і в описі сценарію, і в довідкових матеріалах щодо HA-менеджера, який управляє ресурсом і переносить його на робочу ноду у разі збою.

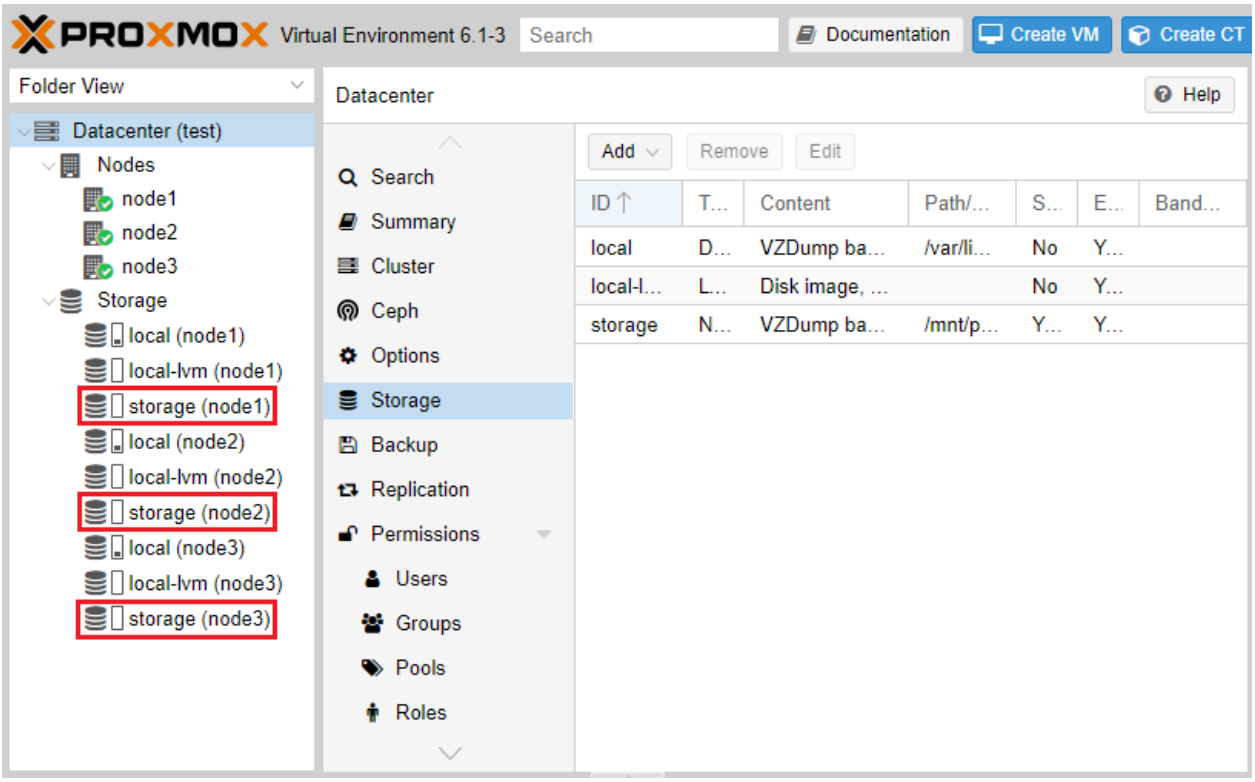

Рисунок 3.11 - Підтвердження підключення NFS-сховища на всіх вузлах кластера (storage присутній для node1-node3).

Після отримання спільного сховища було виконано налаштування High Availability для вибраного ресурсу, що дозволило перейти від "кластер є, але працює лише як централізований інтерфейс" до "кластер забезпечує автоматичне відновлення роботи сервісу після відмови вузла". У демонстраційному прикладі

як керований ресурс використано LXC-контейнер з Ubuntu, однак принцип не залежить від типу ресурсу, оскільки в Proxmox HA може управляти і віртуальними машинами, і контейнерами. На практиці суть процедури полягає в тому, що адміністратор додає ресурс у підсистему HA, після чого HA-стек починає постійно контролювати його стан і координує відновлювальні дії між нодами.

Додавання ресурсу до HA виконується у розділі Datacenter, HA, після чого обирається дія Add і задається ідентифікатор контейнера або ВМ. Додатково встановлюються параметри, які задають межі автоматичних спроб відновлення, зокрема максимальну кількість рестартів і кількість переміщень між нодами. Ці налаштування важливі для практичної експлуатації, оскільки вони визначають, як система поводитиметься у разі “поганого” ресурсу, який не запускається через внутрішню помилку, а не через відмову вузла. У такій ситуації надмірна кількість автоматичних спроб може створити зайве навантаження і ускладнити діагностику, тому обмеження є логічним захистом від циклічного перезапуску.

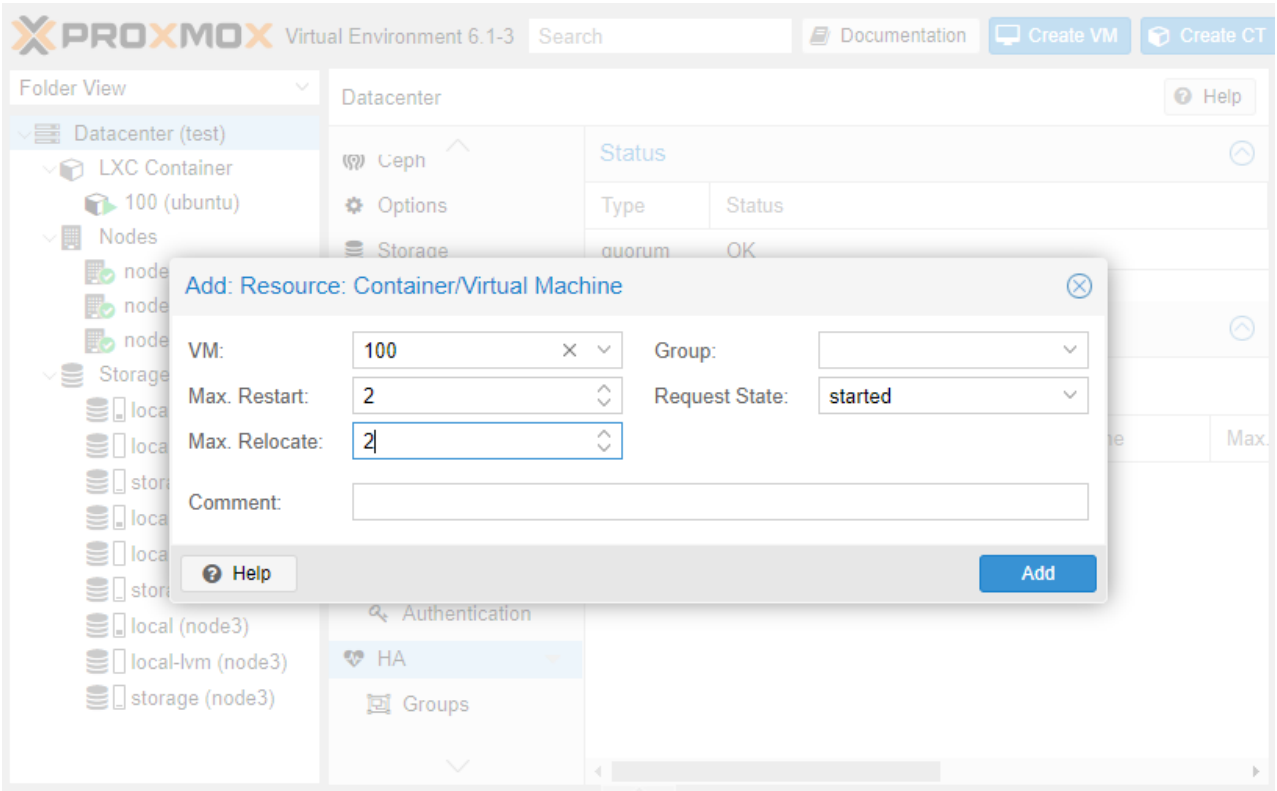

Рисунок 3.12 - Додавання керованого ресурсу до High Availability (вказання ID, Max Restart, Max Relocate).

Після підтвердження додавання ресурс відображається у списку HA, а в секції Status можна бачити стан кворуму та роль нод у HA-стеку. Практично цей екран виконує функцію індикатора “HA увімкнено й реально управляє ресурсом”, оскільки тут видно ідентифікатор ресурсу, його поточний стан, прив’язку до вузла, а також загальний стан кворуму, без якого HA не може коректно приймати рішення. Важливо, що на цьому етапі наявність спільного storage проявляється опосередковано, але принципово: без нього HA зможе лише перезапустити ресурс на тій же ноді, тоді як аварійне переміщення на інший вузол не матиме сенсу, бо дані залишаться на недоступному диску.

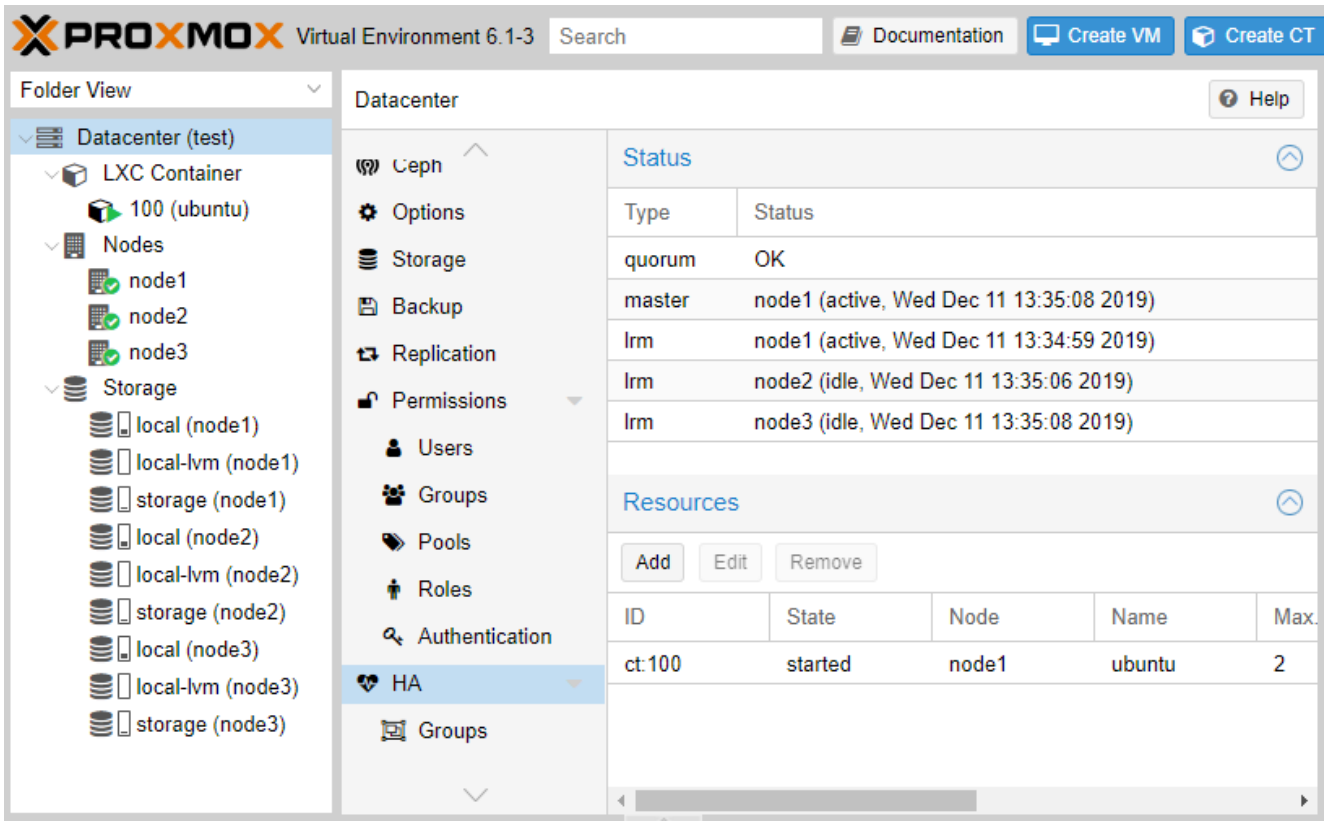

Рисунок 3.13 - Стан HA після додавання ресурсу: відображення керованого контейнера у списку Resources.

Отримані результати були перевірені за допомогою контрольованого “штучного збою”, що є критично важливим для дослідницької частини, адже дозволяє зафіксувати поведінку системи у відмовних режимах. У межах сценарію один із вузлів був вимкнений позаштатно, після чого спостереження

виконувалось з іншої активної ноди через GUI. Система зафіксувала “випадіння” вузла через зміну статусу на недоступний і появу маркерів застарілої мітки часу для відповідних компонентів. Це підтвердило, що механізм моніторингу в складі HA-стеку працює, а кластер коректно ідентифікує недоступність ноди як подію, що вимагає реакції.

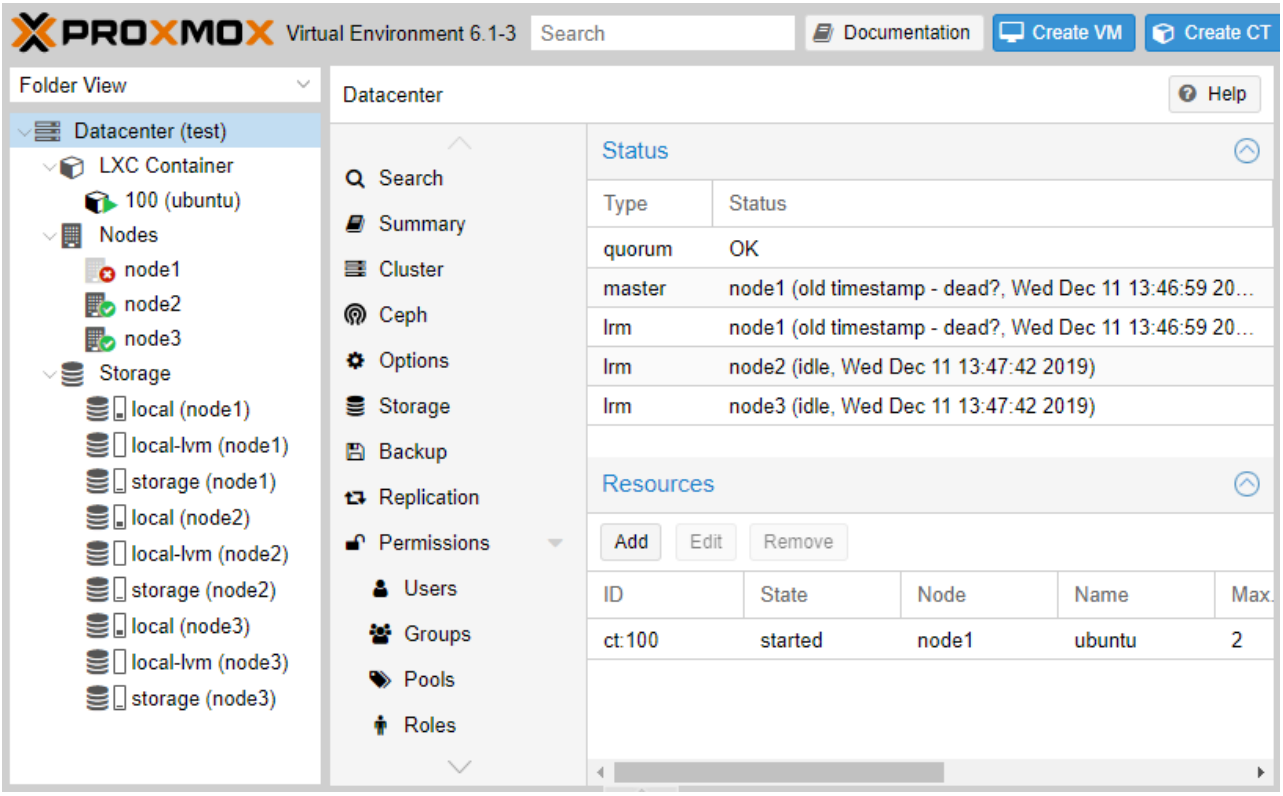

Рисунок 3.14 - Фіксація відмови вузла: кластер відображає недоступність node1 у секції Status.

Найважливішим практичним результатом є те, що HA-менеджер автоматично виконав переназначення вузла для керованого ресурсу і відновив його роботу на іншій ноді в межах приблизно двох хвилин, що відповідає описаній поведінці механізму у реалізованому сценарії. На екрані HA видно, що ресурс перейшов на іншу ноду, при цьому загальний стан кворуму зберігається як валідний, що дозволяє HA приймати рішення і виконувати дії без ручного втручання. У підсумку було підтверджено найцінніший для дата-центру ефект: відмова окремого сервера не призводить до тривалого простою сервісу, а

платформа відновлює працездатність автоматично за рахунок ресурсів кластера і спільного сховища.

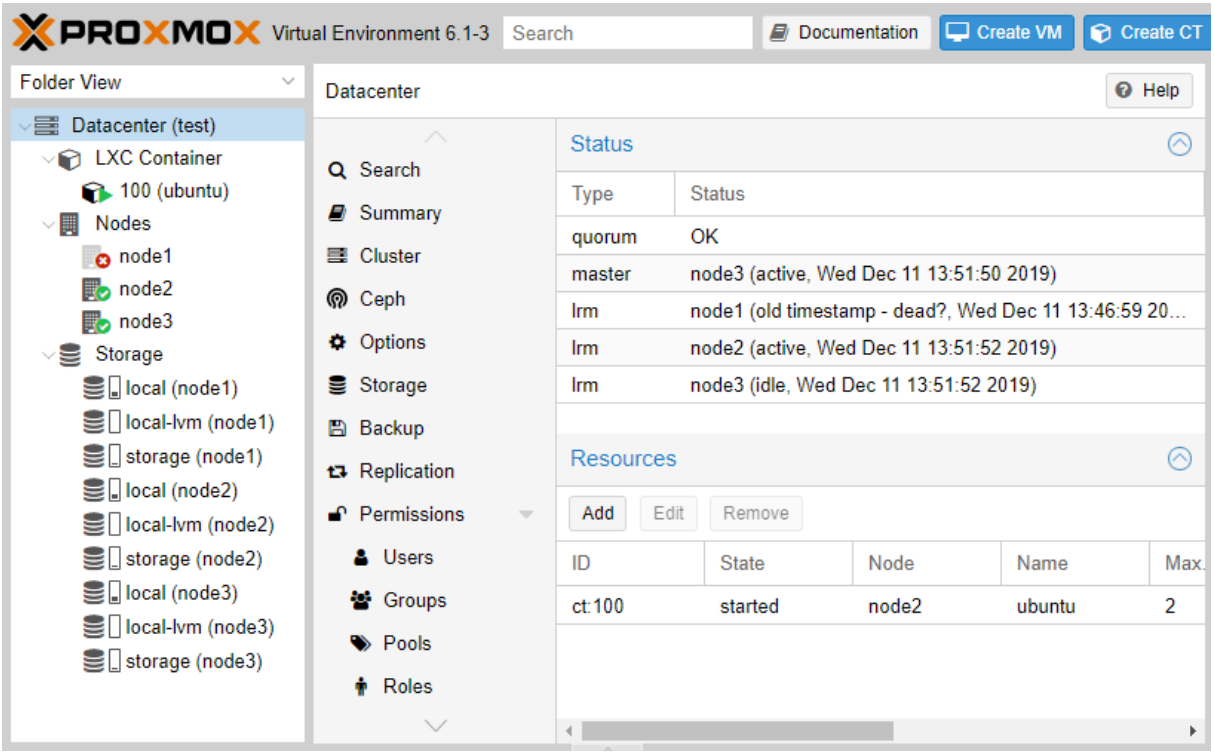

Рисунок 3.15 - Результат спрацювання HA: керований ресурс відновлено на іншій ноді після збою.

Додатково виконано тест на граничний випадок, коли в кластері залишається одна активна нода. Практичний результат цього тесту показує обмеження, яке має суттєве значення для реальної експлуатації: у разі втрати кворуму механізми HA припиняють роботу, оскільки система не може гарантувати узгоджений стан кластера і коректність рішень щодо запуску ресурсів. На екрані явно фіксується повідомлення про відсутність кворуму на залишеній активній ноді, що означає блокування нормального HA-режиму. Це узгоджується з кластерною моделлю Proxmox, де кворум базується на голосуванні, а мінімальна кількість голосів потрібна для прийняття рішень, що пояснюється в матеріалах щодо кластерного менеджера [14].

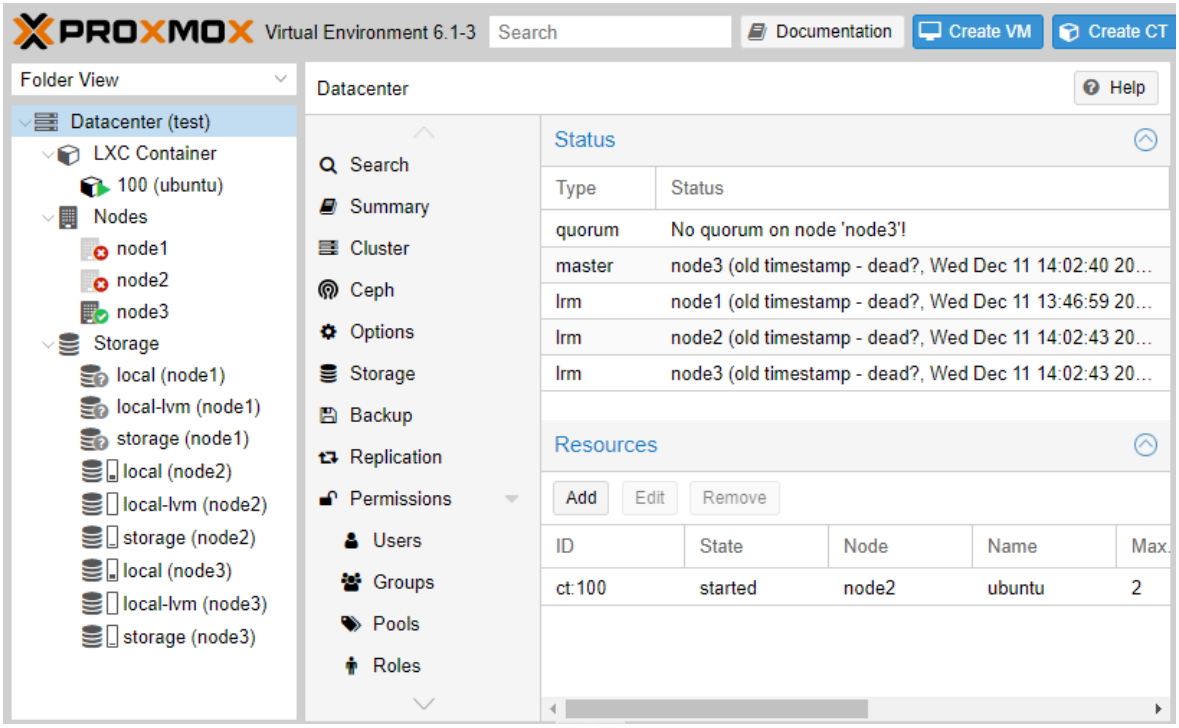

Рисунок 3.16 - Втрата кворуму при відмові двох вузлів: повідомлення про відсутність quorum на node3.

У сценарії було застосовано аварійний прийом примусової зміни очікуваної кількості голосів кластера через команду pvecm expected 1, після чого система відобразила відновлення стану quorum як “OK” і дала змогу механізму HA завершити відновлення ресурсу. Такий крок слід трактувати саме як тимчасове рішення для відновлення керованості у нештатній ситуації, а не як штатну конфігурацію для постійної експлуатації. Практична цінність цього результату полягає в тому, що зафіксовано поведінку системи на межі її коректних режимів: без кворуму HA не працює, з ручним втручанням система може повернутися до керованого стану, але це потребує обережності і розуміння ризиків для узгодженості станів кластера.

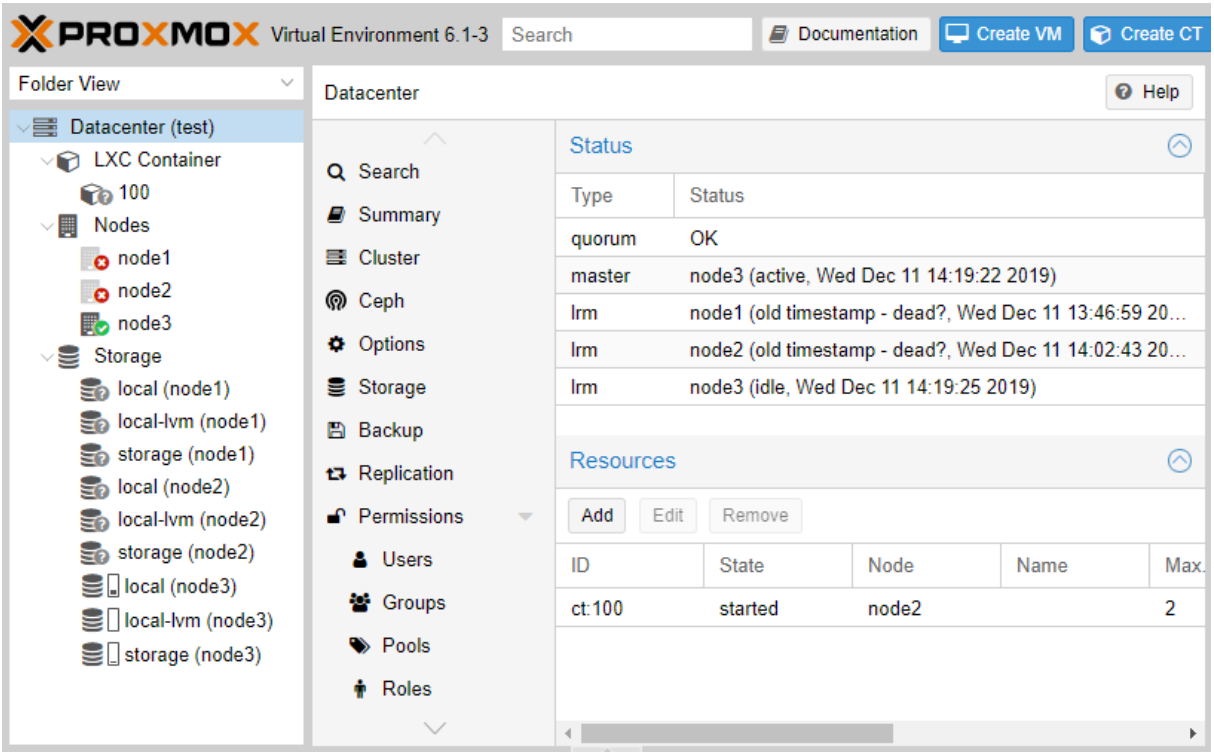

Рисунок 3.17 - Стан кластера після аварійного встановлення очікуваних голосів: quorum відображається як OK.

Фінальним підтвердженням результативності налаштувань стала перевірка фактичного запуску контейнера на новому вузлі. Для цього використано вбудовану консоль Proxmox, де видно, що контейнер доступний і очікує на вхід у систему, а в заголовку вікна зазначено ноду, на якій його запущено. Це важливо з точки зору результатів дослідження, оскільки відображає не лише зміну метаданих у HA-розділі, а реальний перехід виконання сервісу на інший фізичний сервер. Таким чином, отримано експериментальне підтвердження того, що зв’язка “кластер + спільне сховище + HA” дає очікуваний ефект автоматичного відновлення сервісу після відмови вузла з прийнятним часом простою для типових задач дата-центру.

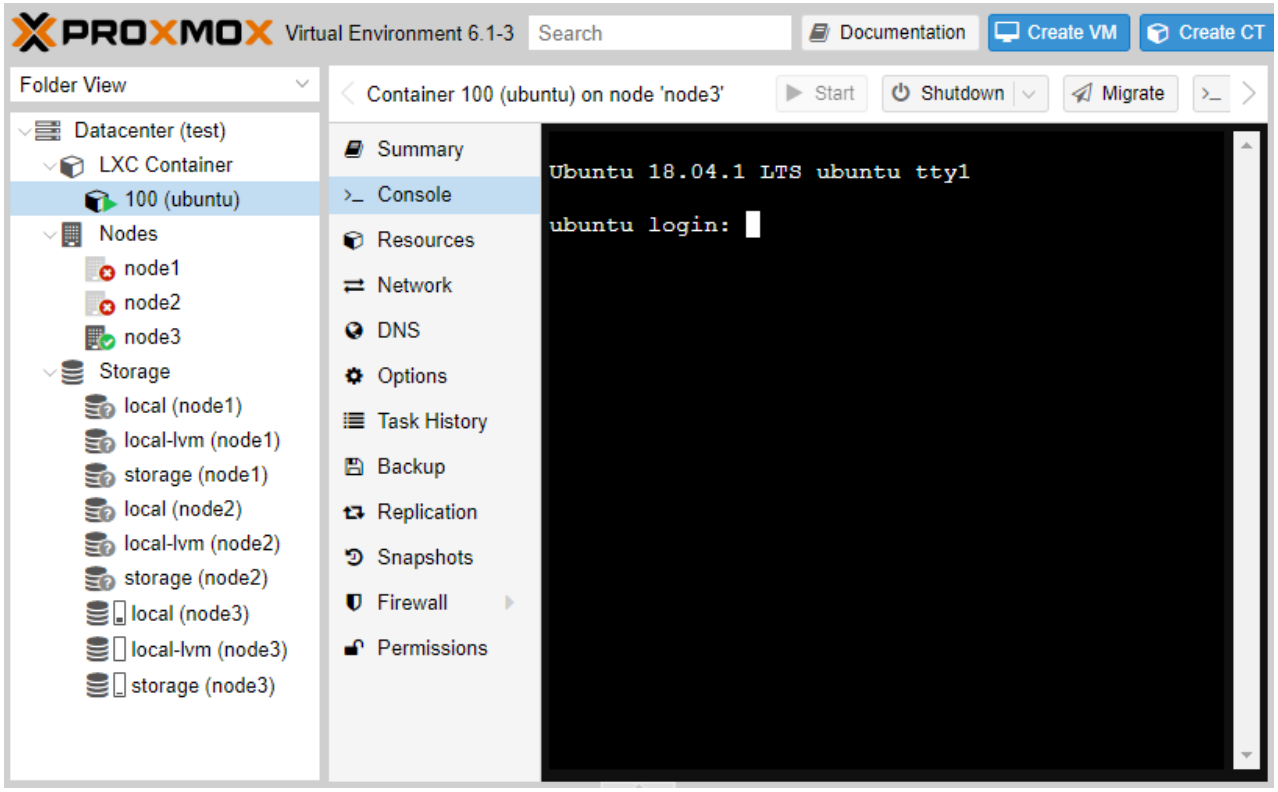

Рисунок 3.18 - Підтвердження запуску ресурсу після переміщення: консоль контейнера на новій ноді.

У підсумку зафіксовано завершений набір практичних результатів. Доведено працездатність підключення спільного NFS-сховища на рівні Datacenter з автоматичною доступністю на всіх вузлах. Підтверджено, що HA-менеджер Proxmox VE починає управляти ресурсом після його додавання, фіксує відмову вузла та виконує автоматичний перезапуск на іншій ноді. Окремо визначено експлуатаційне обмеження, пов’язане з втратою кворуму, що є критичним для проектних рішень щодо мінімальної кількості вузлів у кластері та планування резервування. Саме ці результати формують основу для наступних етапів дослідження, де доцільно переходити до оцінювання стабільності, керованості та ефективності використання ресурсів у різних режимах навантаження.

3.3 Результати оцінювання ефективності використання ресурсів кластера та впливу міграцій на продуктивність сервісів

У межах дослідження після побудови кластера та увімкнення спільного сховища і High Availability було виконано оцінювання того, як змінюється ефективність використання обчислювальних ресурсів при різних підходах до розміщення навантаження, а також як операції керування, зокрема жива міграція, впливають на прикладні показники роботи сервісів. Мета цього підрозділу полягає у фіксації результатів, які можна безпосередньо пов’язати з оптимізацією дата-центру: підвищенням утилізації ресурсів, зменшенням простоїв, стабілізацією якості сервісу та зниженням ризиків, пов’язаних із технічними роботами. Для збору метрик у практичних умовах доцільно використовувати комбінацію системного експорту метрик вузлів і збору показників з боку Proxmox API або експортерів, які читають інформацію з Proxmox VE та віддають її у форматі, зручному для Prometheus. Системний рівень закривається Node Exporter, який надає метрики ядра, процесора, пам’яті, дисків і мережі, а специфічні показники Proxmox можна збирати через Prometheus Proxmox VE Exporter або через REST API Proxmox VE.

Важливо зазначити, що сам Proxmox VE також підтримує відправлення статистики на зовнішні сервери метрик, що може використовуватися як альтернатива або доповнення до підходу з Prometheus, залежно від обраної архітектури моніторингу. У цьому дослідженні логіка вимірювань розглядається саме як результативна частина, де метрики слугують інструментом порівняння режимів експлуатації кластера.

Дослідницький підхід було побудовано навколо двох типових режимів роботи, які на практиці зустрічаються в дата-центрах. Перший режим відповідає ситуації, коли навантаження розподілене між вузлами відносно рівномірно і кластер працює у стані балансування. Другий режим імітує консолідацію: частина віртуальних машин і контейнерів переноситься на меншу кількість вузлів у період низького попиту, а один із серверів залишається майже без навантаження, що створює можливість для енергозбереження або виконання планових робіт без помітного впливу на сервіси. Обидва режими мають сенс для

оптимізації, однак формують різний профіль ризиків і по-різному впливають на продуктивність та поведінку дискової підсистеми під час міграцій.

Щоб результат було зручно оформити у вигляді графіків, метрики агрегуються в часові ряди з фіксованим кроком дискретизації. У практиці це може бути інтервал 10–30 секунд, проте для демонстрації у тексті роботи нами було використано хвилинний крок, який не перевантажує графіки і зберігає наочність. Основними показниками, які оцінювалися, є завантаженість CPU по вузлах, використання оперативної пам’яті, затримка дискових операцій як індикатор стану I/O, а також прикладний показник якості сервісу у вигляді p95 часу відповіді. Концептуально саме p95 є зручним для фіксації впливу міграцій, оскільки середнє значення часто “маскує” короткі деградації, тоді як 95-й показує, що відчуває користувач у пікові моменти.

Нижче наведено графіки, які демонструють формат подання результатів і взаємозв’язок між режимом розміщення навантаження та поведінкою показників у трьох сценаріях.

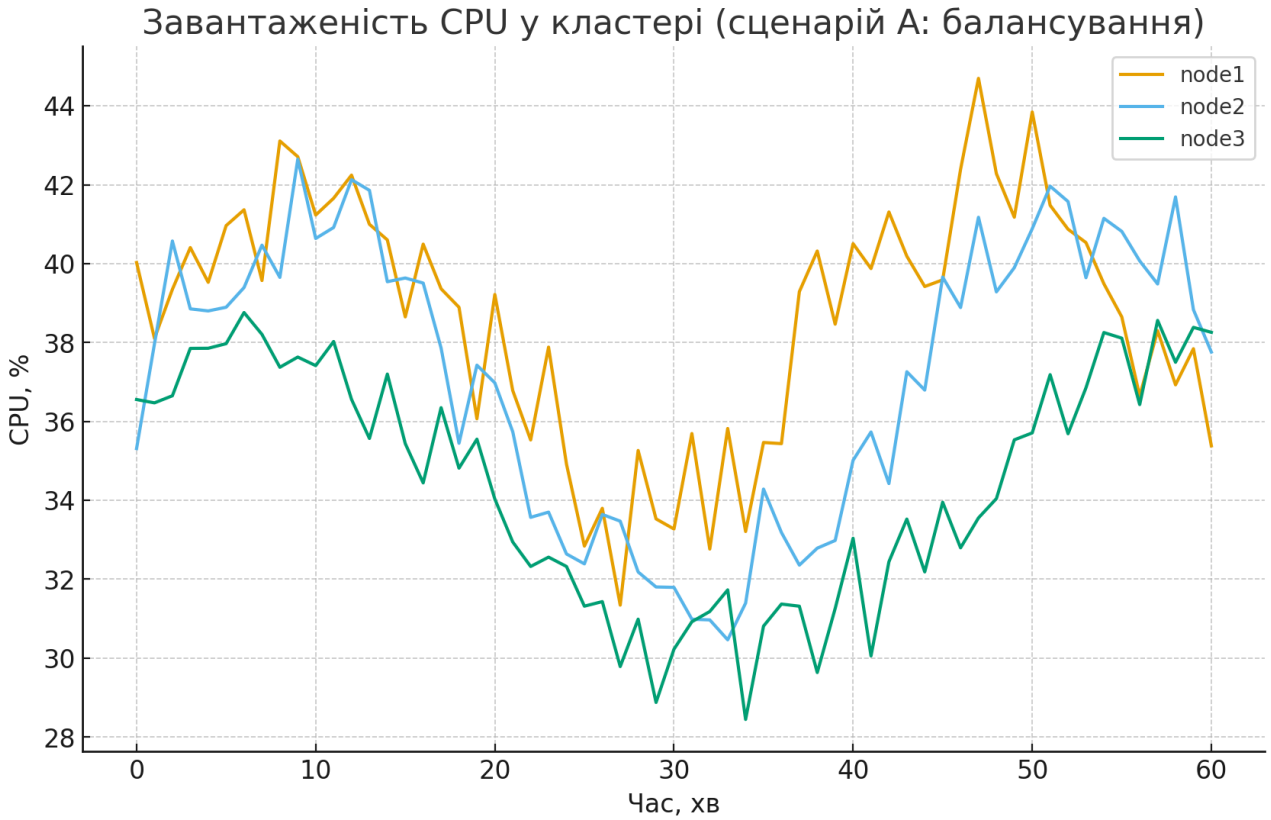

Рисунок 3.19 – Завантаженість CPU у кластері (сценарій A: балансування навантаження).

Отримана динаміка демонструє, що при балансованому підході навантаження розподіляється між трьома вузлами без різких піків, а коливання завантаженості CPU залишаються в межах, які не створюють передумов для черг у планувальнику та для росту затримок на прикладному рівні. Практично це важливо тим, що навіть при зростанні кількості запитів кластер має запас, а рішення про міграцію можна приймати без ризику перевантажити окремий вузол. У цьому режимі оптимізація досягається через стабільність і прогнозованість, а також через можливість уникати ручних дій завдяки централізованому контролю стану кластера.

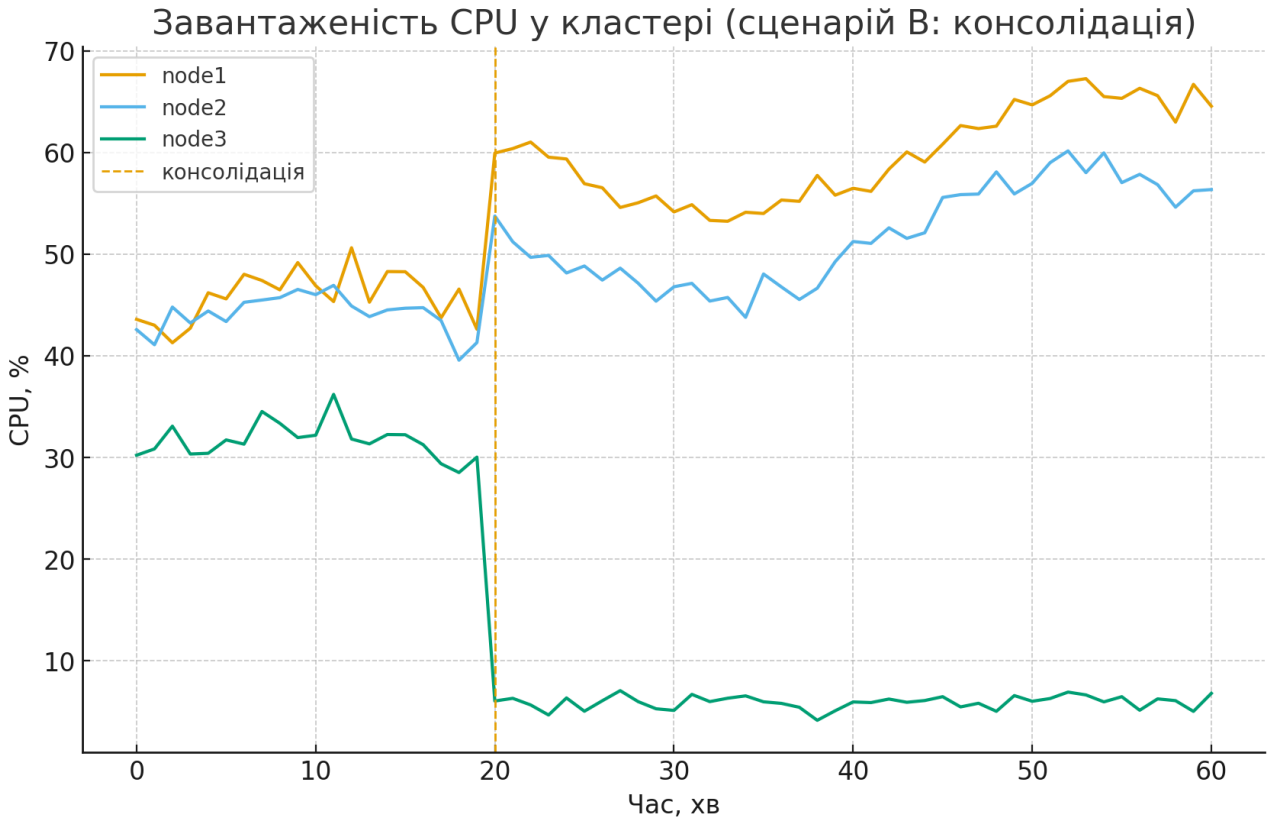

Рисунок 3.20 – Завантаженість CPU у кластері (сценарій B: консолідація; момент консолідації позначено вертикальною лінією).

У режимі консолідації після контрольного моменту перенесення частини навантаження на дві ноди спостерігається закономірний ріст завантаженості CPU на активних вузлах і різке зниження активності на третьому сервері. Це відображає цільовий ефект консолідації: підвищення утилізації ресурсів на

меншій кількості фізичних систем, що створює потенціал для оптимізації витрат на енергоспоживання та обслуговування інфраструктури. Разом з тим цей режим робить систему чутливішою до раптових піків, оскільки запас продуктивності концентрується на меншій кількості вузлів. У практичній експлуатації це означає, що консолідація має виконуватися за зрозумілими правилами і за наявності прозорих тригерів від моніторингу, які повертають інфраструктуру у балансований режим при зростанні навантаження.

Окремим предметом аналізу стала жива міграція як інструмент оптимізації, адже на практиці її застосовують і для балансування, і для виведення вузла на технічне обслуговування, і для підготовки до консолідації. Хоча міграція з точки зору користувача часто сприймається як “прозора” процедура, вона створює короткочасний тиск на мережу і дискову підсистему, а в разі спільних сховищ може спричиняти піки затримок I/O. Саме тому в рамках результатів важливо показувати не лише завантаження CPU, а й затримку дискових операцій у момент міграції.

Рисунок 3.21 – Затримка дискових операцій під час живої міграції (період міграції виділено на графіку).

На графіку видно, що під час міграції виникає короткий сплеск затримки дискових операцій, після чого показник повертається до базового рівня. Така поведінка є типовою: міграція генерує додатковий трафік і збільшує кількість операцій читання і запису, а також створює конкуренцію за ресурси між потоками обслуговування дисків і мережі. Практичний висновок полягає в тому, що жива міграція є корисним інструментом оптимізації, але її бажано запускати у контрольовані часові вікна або при достатньому ресурсному запасі, а також відстежувати I/O метрики у момент виконання операції, аби переконатися, що деградація не виходить за прийнятні межі.

Для зв’язку інфраструктурних метрик із відчутною якістю сервісу було розглянуто p95 часу відповіді, оскільки цей показник у середовищах веб-сервісів та API добре відображає випадки коротких погіршень. Високий p95 означає, що у частини запитів час обробки істотно зріс, навіть якщо середнє значення лишається стабільним.

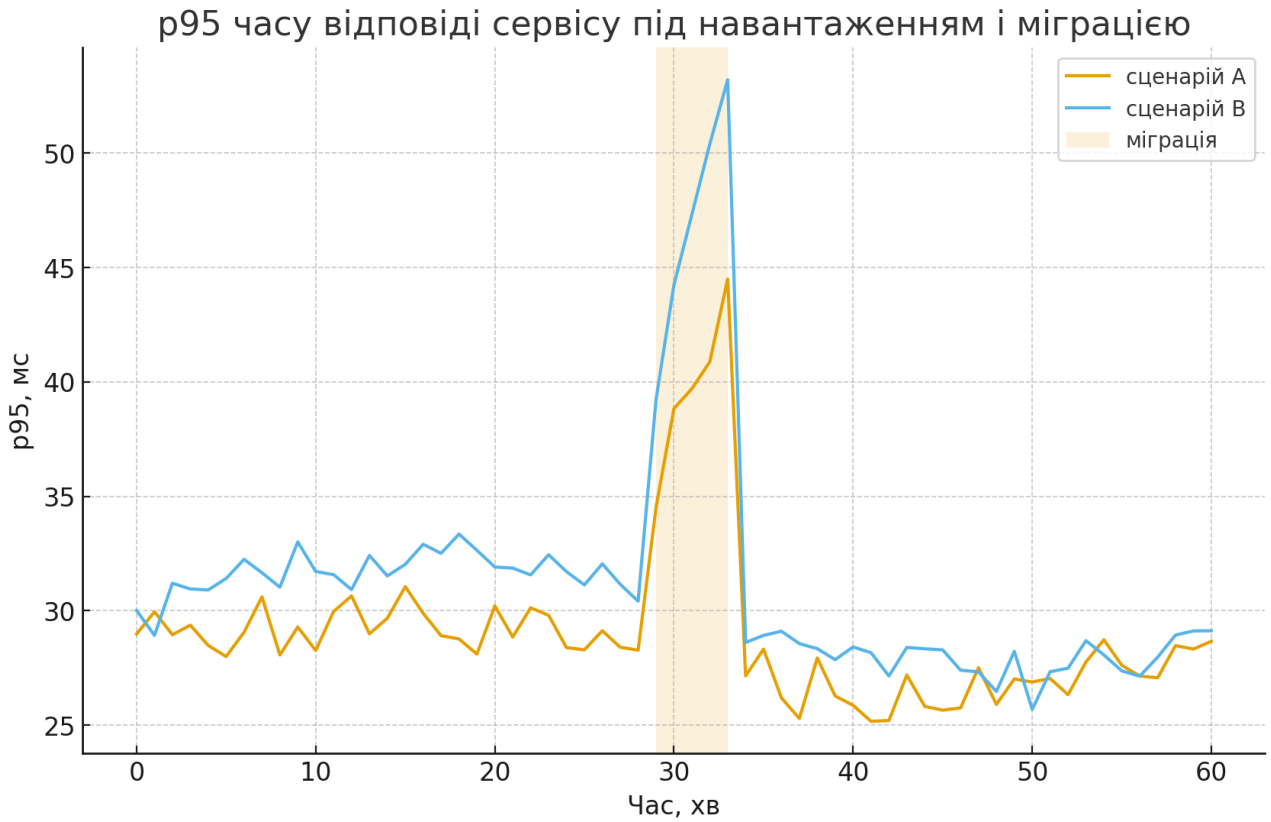

Рисунок 3.22 – p95 часу відповіді сервісу під навантаженням і під час міграції.

Зіставлення графіків показує, що у момент міграції p95 має тенденцію до зростання, що узгоджується з піком затримок I/O. Водночас сам факт повернення показників до стабільного рівня після завершення міграції демонструє працездатність підходу: міграція як операція керування не руйнує сервіс і не створює довготривалих наслідків, якщо інфраструктура має достатні ресурси і правильно організоване спільне сховище. На рівні оптимізації це означає, що керовані міграції можуть бути регулярним механізмом експлуатації кластера, але вони мають супроводжуватися моніторингом і контрольними порогами, щоб не переносити ресурси у невдалий момент.

3.4 Узагальнення результатів вирішення задачі та оцінка досягнутого ефекту оптимізації дата-центру

Отримані в межах роботи результати дозволяють зробити висновок, що поставлену задачу оптимізації роботи дата-центру на основі віртуалізації обчислювальних ресурсів вирішено комплексно, тобто на рівні інфраструктурної архітектури, процедур експлуатації та інструментальної бази контролю якості сервісу. Побудований кластер Proxmox VE забезпечив перехід від моделі ізольованих серверів до єдиного керованого середовища, у якому ресурси CPU, RAM, мережа та сховища сприймаються як спільний пул і можуть розподілятися відповідно до поточних потреб. Такий результат важливий не лише як факт інсталяції платформи, а як практично реалізована основа для консолідації навантажень, зменшення простоїв і підвищення керованості, що відповідає сучасним підходам до побудови віртуалізованих дата-центрів і хмарних середовищ.

З позиції технічного виконання ключовим досягненням стало впровадження кластерного механізму Proxmox VE із коректним кворумом, що підтверджувалося станом Quorate та відображенням усіх нод як Online в єдиному інтерфейсі Datacenter. Саме кворум є критичною умовою для узгоджених рішень у кластері, оскільки без нього неможливо гарантувати коректність операцій

керування ресурсами, зокрема запуску, зупинки або переміщення сервісів. Експериментальна перевірка сценаріїв із частковою втратою вузлів показала, що втрата кворуму призводить до обмеження функціональності кластерних підсистем і потребує окремої уваги під час проектування кількості вузлів та резервування. На практиці це означає, що оптимізація дата-центру не зводиться до зменшення кількості серверів, а потребує балансу між економією ресурсів і збереженням умов для стійкого керування.

Другою критичною складовою вирішення задачі стало підключення спільного сховища, яке забезпечило винесення дискових даних віртуальних машин і контейнерів за межі локальних дисків окремого вузла. Це створило технічну передумову для відмовостійкого перезапуску сервісів на іншій ноді, а також для застосування живої міграції як інструменту оптимізації розміщення навантаження. У реалізованому сценарії використання мережевого NFS як спільного storage дозволило швидко отримати загальний простір даних для всіх вузлів і перевірити працездатність на рівні інтерфейсу та експлуатації. Разом із тим логіка рішення не обмежується лише NFS, оскільки в промислових середовищах часто застосовують також розподілені сховища або SAN, а Proxmox VE підтримує різні моделі інтеграції storage відповідно до технічних можливостей майданчика. Отже, отриманий результат має переносимість: зберігаються принципи, а тип конкретного storage може змінюватися без руйнування загальної архітектури.

Найбільш показовим ефектом оптимізації стало впровадження High Availability для вибраних ресурсів та перевірка автоматичного відновлення роботи сервісів після відмови вузла. Практичний тест із вимкненням ноди продемонстрував, що HA-менеджер здатний зафіксувати збій, переназначити ресурс і відновити його запуск на доступному вузлі без ручного втручання [9; 12]. Це означає, що при коректних умовах кворуму і спільного сховища відмова окремого фізичного сервера не перетворюється на тривалий простій сервісу, а стає подією, яку кластер обробляє штатними механізмами. Для дата-центру це напряму трансформується в практичну оптимізацію, оскільки зменшується

потреба у надмірному апаратному резервуванні на рівні кожного сервісу, а стабільність досягається на рівні платформи, яка керує ресурсами. При цьому результати експериментів підкреслили важливість правильного проектування кластера, адже HA у Proxmox VE опирається на кворум і не може вважатися заміною продуманій архітектурі резервування.

Окремий пласт отриманих результатів стосується керованості робіт з обслуговування і експлуатаційної гнучкості, які безпосередньо впливають на простій і витрати на підтримку інфраструктури. Наявність кластера з можливістю міграції означає, що планові роботи на вузлі можна виконувати без зупинки сервісів, попередньо перемістивши їх на інші ноди. Такий підхід знижує організаційні витрати на виконання оновлень, заміну обладнання чи оптимізацію конфігурацій, оскільки не вимагає “вікон простою” для кожного сервісу окремо. У підрозділах 3.1 і 3.2 це проявилося як практична здатність керувати ресурсами у межах єдиного дата-центру і переносити запуск керованого ресурсу у відповідь на відмову, що є по суті двома сторонами однієї властивості: платформа бере на себе узгоджені дії керування.

З позиції ефективності використання ресурсів суттєвим результатом стала можливість порівнювати режими балансування і консолідації на основі метрик. Під час аналізу показано, що балансований режим дає рівномірну утилізацію CPU між вузлами і формує запас продуктивності для пікових значень навантаження. Натомість консолідація підвищує завантаженість двох вузлів і знижує активність третього, що створює простір для оптимізації енергоспоживання та для технічних робіт на вивільненому сервері. Водночас консолідація підвищує чутливість до раптових піків і потребує чітких правил повернення в балансований режим за сигналами моніторингу. У роботі цей зв’язок показано через результати в часових рядах для CPU та через інтерпретацію тенденцій, які є типовими для віртуалізованих середовищ, де ефект оптимізації проявляється через зміну профілю навантаження на фізичних вузлах.

Окремо зафіксовано вплив операцій живої міграції на дискову підсистему і на прикладний показник якості сервісу у вигляді p95 часу відповіді. Результати демонструють, що міграція може створювати короткочасне зростання затримок I/O, яке відображається у перцентилях часу відповіді, навіть якщо середні значення лишаються стабільними. Практичний висновок із цих спостережень полягає в тому, що міграція є робочим інструментом оптимізації для технічного обслуговування і перекладання навантажень, але її не слід виконувати безконтрольно. Вона має супроводжуватися моніторингом, а порогові значення для I/O та p95 повинні бути відомі заздалегідь, щоб у разі деградації можна було змінити стратегію, наприклад відкласти міграцію або виконати її в інший часовий проміжок. Можливість отримувати такі метрики забезпечується інструментальною базою на основі Node Exporter, Proxmox API, експортера Proxmox для Prometheus і механізмів інтеграції Proxmox із зовнішніми серверами метрик.

Важливим результатом, який має методичне значення для подальшої експлуатації, стала побудова відтворюваного підходу до вимірювання і документування ефекту оптимізації. Оптимізація дата-центру часто оцінюється узагальненими твердженнями про покращення ефективності, однак у цій роботі показано, як перетворити це на вимірювані показники та оформити їх у вигляді графіків, придатних для порівняння режимів. За наявності повторного прогону тестів із реальними даними конкретного майданчика можна отримувати серії метрик і будувати порівняння за однаковими метриками, що дає підстави для обґрунтованих управлінських рішень щодо розширення кластера, зміни типу сховища або корекції політик міграції та консолідації. Такий підхід підсилює практичну цінність роботи, оскільки результати не залишаються разовим експериментом, а стають основою для системного управління інфраструктурою.

Отже, підсумковий ефект вирішення задачі проявляється у трьох площинах. На архітектурному рівні створено кластерне середовище з централізованим керуванням, що дозволяє ефективно розподіляти ресурси та масштабувати інфраструктуру. На рівні надійності реалізовано спільне сховище

і High Availability, що забезпечує автоматичне відновлення сервісів при відмові вузлів у межах умов кворуму та коректної конфігурації. На рівні експлуатаційної оптимізації сформовано інструментальну базу для вимірювань і доведено, що політики балансування, консолідації та міграції можуть оцінюватися через метрики CPU, I/O і p95, що дозволяє не інтуїтивно, а на підставі даних формувати рішення щодо режимів роботи дата-центру. Сукупно ці результати підтверджують, що віртуалізація обчислювальних ресурсів у поєднанні з кластеризацією та сучасними підходами до моніторингу є дієвим інструментом оптимізації роботи дата-центрів, який забезпечує як підвищення ефективності, так і стійкість сервісів у реальних експлуатаційних умовах.

РОЗДІЛ 4.
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Розробка логіко-імітаційної моделі виникнення травм і аварій

Методикою оцінки рівня небезпеки робочих місць, машин, виробничих процесів та окремих виробництв передбачено пошук об'єктивного критерію рівня небезпеки для конкретного об'єкта [19]. Таким показником вибрана ймовірність виникнення аварії, травми залежно від явища, що досліджується.

Для побудови логіко-імітаційної моделі процесу, формування і виникнення аварії та травми в процесі створення мікрокліматичних умов у приміщенні оцінюють відповідні небезпечні події. Кожній їх них присвоїмо ймовірність виникнення:

Шифр	Назва події	Ймовірність
P_1	Відсутність захисного заземлення	0,02
P_2	Пошкодження захисного заземлення	0,04
P_3	Спрацювання складових захисту	0,1
P_4	Неправильна експлуатація захисту	0,02
P_5	Відсутність профілактичних заходів	0,2
P_6	Відсутність захисного щита	0,12
P_7	Недотримання правил вибору взуття	0,15
P_8	Незнання правил техніки безпеки	0,1
P_9	Відсутність засобів індивідуального захисту	0,2
P_{10}	Легковажність	0,08

На основі наведених подій будуємо матрицю логічних взаємозв'язків між окремими пунктами, графічна інтерпретація якої зображено на рис. 4.1.

Розрахуємо ймовірності виникнення подій, що формують логіко-імітаційну модель процесів створення мікрокліматичних умов. Розглянемо травмонебезпечну ситуацію, що виникає за умови роботи працівників із електронебезпекою.

Підставивши дані ймовірностей базових подій у формулу, отримаємо ймовірність події 13: $P_{13} = 0{,}2 + 0{,}4 - 0{,}2 \cdot 0{,}4 = 0{,}0592$.

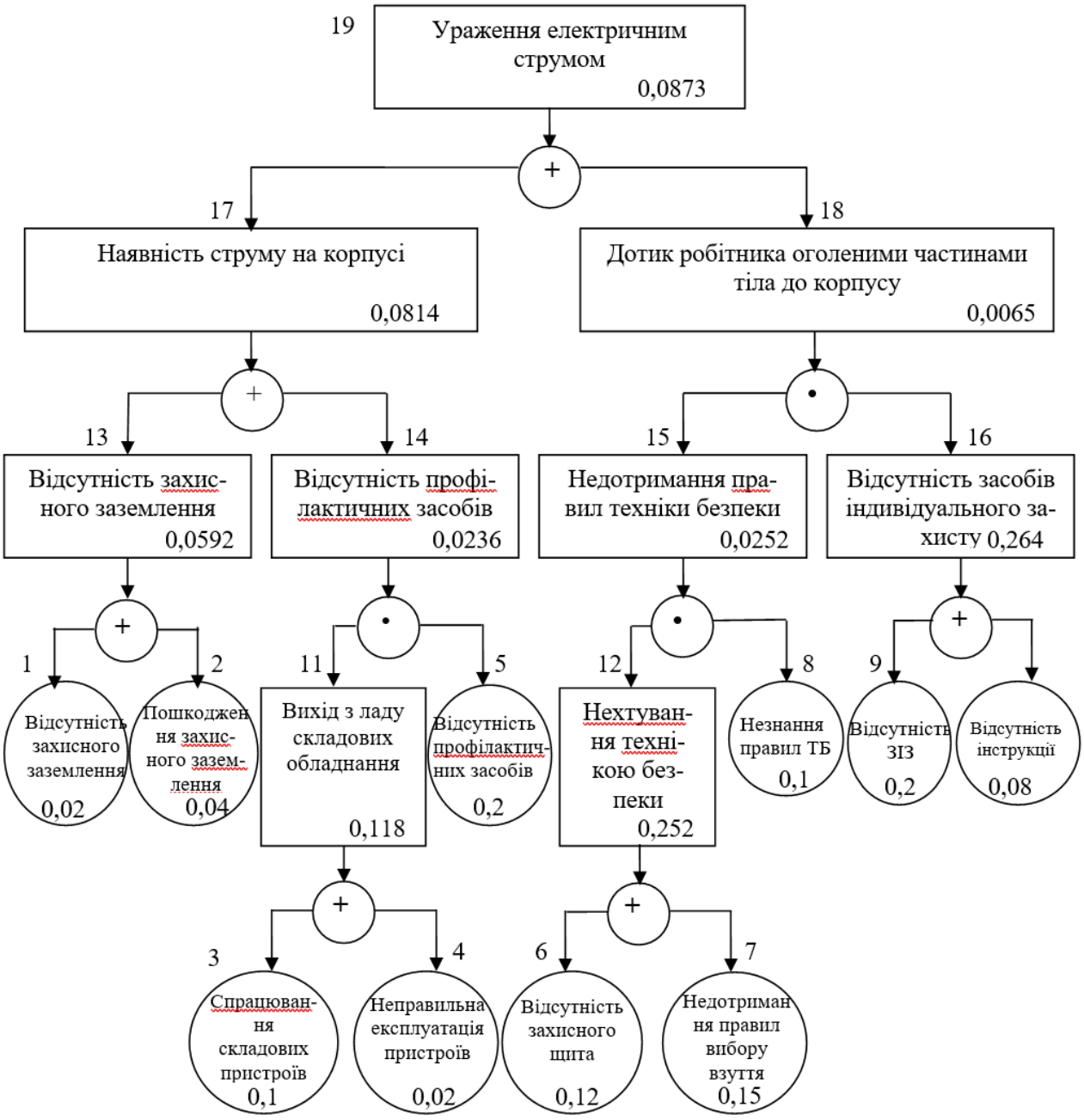

Рис. 4.1. Матриця логічних взаємозв'язків між окремими подіями травмонебезпечної ситуації [19]

Аналогічно визначаємо ймовірність інших подій:

$$P_{11} = P_4 + P_5 - P_4P_5 = 0,3 + 0,4 - 0,3 \cdot 0,4 = 0,118.$$

$$P_{12} = P_6 + P_7 - P_6P_7 = 0,3 + 0,5 - 0,3 \cdot 0,5 = 0,252.$$

$$P_{16} = P_9 + P_{10} - P_9P_{10} = 0,2 + 0,15 - 0,2 \cdot 0,15 = 0,264.$$

$$P_{14} = P_{11} \cdot P_5 = 0,118 \cdot 0,2 = 0,0236.$$

$$P_{15} = P_{12} \cdot P_8 = 0{,}252 \cdot 0{,}1 = 0{,}0252.$$

$$P_{17} = P_{13} + P_{14} - P_{13} \cdot P_{14} = 0{,}592 + 0{,}0236 - 0{,}0592 \cdot 0{,}0236 = 0{,}0814.$$

$$P_{18} = P_{15} \cdot P_{16} = 0{,}264 \cdot 0{,}0252 = 0{,}0065.$$

$$P_{19} = P_{17} + P_{18} - P_{17} \cdot P_{18} = 0{,}0065 + 0{,}0814 - 0{,}0065 \cdot 0{,}0814 = 0{,}0873.$$

Таким чином, ймовірність перекидання машини та наслідкового виникнення травми працівника є досить мала і становить – $P_{19} = 0{,}0873$.

4.2. Планування заходів із покращення умов праці

До заходів щодо покращення умов праці належать всі види діяльності, спрямовані на попередження, нейтралізацію або зменшення негативної дії шкідливих і небезпечних виробничих факторів на працівників.

Рівень умов праці оцінюють порівнянням за фактичними і нормативними значеннями узагальнених (групових) показників.

Заходи щодо поліпшення умов праці здійснюють з метою створення безпечних умов праці шляхом:

- доведення до нормативного рівня показників виробничого середовища за елементами умов праці;
- захисту працівників від дії небезпечних і шкідливих виробничих факторів.

До показників ефективності заходів щодо поліпшення умов праці належать:

а) зміни стану умов праці:

- зміна кількості засобів виробництва, приведених у відповідність до вимог стандартів безпеки праці;
- покращання санітарно-гігієнічних показників;
- покращання психофізичних показників, зменшення фізичних і нервово-психічних навантажень, в т.ч. монотонних умов праці;

- покращання естетичних показників, раціональне компонування робочих місць і впорядкування робочих приміщень;

б) соціальні результати заходів:

- збільшення кількості робочих місць, що відповідають нормативним вимогам;
- зниження рівня виробничого травматизму;
- зменшення кількості випадків професійних захворювань;
- зменшення плинності кадрів через незадовільні умови праці;
- престиж та задоволення працею.

Отже, на покращення охорони праці потрібно виділити кошти на відновлення вентиляційних систем у ремонтних майстернях, естетично оформити приміщення офісу, відновити кабінет з охорони праці, поновити протипожежний інвентар.

4.3. Безпека в надзвичайних ситуаціях

Актуальність проблеми природно-техногенної безпеки для населення і території, зумовлена зростанням втрат людей, що спричиняється небезпечними природними явищами, промисловими аваріями та катастрофами. Ризик надзвичайних ситуацій природного та техногенного характеру невпинно зростає, тому питання захисту цивільного населення від надзвичайних ситуацій на сьогодні є дуже важливе [20].

У системі цивільної оборони окремого господарства необхідно забезпечити захист населення таким чином:

Укриття в захисних спорудах, якому підлягає усе населення відповідно до приналежності, досягається створенням фонду захисних споруд.

Евакуаційні заходи, які проводяться в містах та інших населених пунктах, які мають об’єкти підвищеної небезпеки, а також у воєнний час, основним способом захисту населення є евакуація і розміщення його у позаміській зоні.

Медичний захист проводиться для зменшення ступеня ураження людей, своєчасного надання допомоги постраждалим та їх лікування, забезпечення епідеміологічного благополуччя в районах надзвичайних ситуацій.

Радіаційний і хімічний захист включає заходи щодо виявлення і оцінки радіаційної та хімічної обстановки, організацію і здійснення дозиметричного та хімічного контролю, розроблення типових режимів радіаційного захисту, забезпечення засобами індивідуального захисту, організацію і проведення спеціальної обробки.

Евакуаційні заходи, які проводяться в містах та інших населених пунктах, які мають об’єкти підвищеної небезпеки, а також у воєнний час, основним способом захисту населення є евакуація і розміщення у позаміській зоні [19,20].

РОЗДІЛ 5.
ВИЗНАЧЕННЯ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ ІНСТРУМЕНТАРІЮ ВІРТУАЛІЗАЦІЇ ДЛЯ ОПТИМІЗАЦІЇ РОБОТИ ДАТА-ЦЕНТРІВ

Ефективність запропонованого інструментарію доцільно оцінювати за сукупністю показників, які відображають економічний ефект від консолідації обчислювальних ресурсів, зміну енергоспоживання, зниження трудових витрат на адміністрування, а також вартість розроблення і впровадження рішення. У межах цього розділу наведено методику розрахунку та приклад обчислення показників для типового сценарію експлуатації кластера Proxmox VE та супутнього інструментарію моніторингу й регламентів адміністрування. Вихідні числові значення прийнято як репрезентативні для лабораторного або малого продуктивного майданчика, а сама методика не залежить від конкретних цифр і може бути повторена для іншої конфігурації.

Для розрахунків прийнято, що до впровадження віртуалізації експлуатувалося п’ять фізичних серверів із середньою потужністю споживання одного сервера $\boldsymbol{P_1 = 0,32}$ кВт. Після впровадження кластера навантаження було консолідовано на трьох вузлах із середньою потужністю одного активного вузла $\boldsymbol{P_2 = 0,38}$ кВт, а також використано мережеве сховище з середньою потужністю $\boldsymbol{P_s = 0,12}$ кВт. Режим роботи цілодобовий, тобто $\boldsymbol{T = 8760}$ год/рік. Для врахування загальнодатацентрових витрат на охолодження та інженерну інфраструктуру застосовано коефіцієнт енергоефективності майданчика $\boldsymbol{PUE = 1,6}$. Тариф на електроенергію прийнято умовно на рівні $\boldsymbol{t_e = 7,0}$ грн/кВт\cdotргод, що відповідає типовому комерційному діапазону вартості з урахуванням розподілу.

Річне енергоспоживання IT-частини (без урахування $\boldsymbol{PUE}$) визначається за формулою

$$\boldsymbol{W}_{\text{IT}} = \boldsymbol{P}_{\text{IT}} \cdot \boldsymbol{T},$$

де $\boldsymbol{P}_{\text{IT}}$це сумарна середня потужність IT-обладнання, кВт, $\boldsymbol{T}$це час роботи, год. Повне річне енергоспоживання майданчика з урахуванням інженерної інфраструктури оцінюється як

$$\boldsymbol{W}_{\text{DC}} = \boldsymbol{PUE} \cdot \boldsymbol{W}_{\text{IT}}.$$

Тоді вартість електроенергії за рік становить

$$\boldsymbol{C_e} = \boldsymbol{W}_{\text{DC}} \cdot \boldsymbol{t_e}.$$

До впровадження сумарна потужність IT-частини дорівнює

$$\boldsymbol{P}_{\text{IT},0} = \mathbf{5} \cdot \mathbf{0,32} = \mathbf{1,60}\ \text{кВт},$$

а після впровадження

$$\boldsymbol{P}_{\text{IT},1} = \mathbf{3} \cdot \mathbf{0,38} + \mathbf{0,12} = \mathbf{1,26}\ \text{кВт}.$$

Звідси річне енергоспоживання з урахуванням $\boldsymbol{PUE}$для базового стану:

$$\boldsymbol{W}_{\text{DC},0} = \mathbf{1,6} \cdot \mathbf{1,60} \cdot \mathbf{8760} = \mathbf{22425,6}\ \text{кВт\cdotprгод},$$

а для стану після впровадження:

$$\boldsymbol{W}_{\text{DC},1} = \mathbf{1,6} \cdot \mathbf{1,26} \cdot \mathbf{8760} = \mathbf{17660,16}\ \text{кВт\cdotprгод}.$$

Річна економія електроенергії:

$$\boldsymbol{\Delta W} = \boldsymbol{W}_{\text{DC},0} - \boldsymbol{W}_{\text{DC},1} = \mathbf{4765,44}\ \text{кВт\cdotprгод}.$$

Річна економія коштів на електроенергії:

$$\boldsymbol{\Delta C_e} = (\boldsymbol{W}_{\text{DC},0} - \boldsymbol{W}_{\text{DC},1}) \cdot \boldsymbol{t_e} = \mathbf{4765,44} \cdot \mathbf{7,0} = \mathbf{33358,08}\ \text{грн}.$$

Відносний показник енергетичної ефективності, як частка зниження витрат електроенергії, визначається:

$$\boldsymbol{\eta_E} = \frac{\boldsymbol{\Delta W}}{\boldsymbol{W}_{\text{DC},0}} \cdot \mathbf{100\%} = \frac{\mathbf{4765,44}}{\mathbf{22425,6}} \cdot \mathbf{100\%} \approx \mathbf{21,25\%}.$$

За потреби можна додатково оцінити екологічний ефект як скорочення викидів $\boldsymbol{CO_2}$за емісійним коефіцієнтом $\boldsymbol{k_{CO_2}}$(умовно $\mathbf{0,4}$ кг/кВт\cdotpгод):

$$\boldsymbol{\Delta CO_2 = \Delta W \cdot k_{CO_2} = 4765,44 \cdot 0,4 \approx 1906}\ \text{кг/рік},$$

що відповідає приблизно $\mathbf{1,9}$ т/рік.

Наступною складовою ефективності є трудомісткість експлуатації та пов'язані з нею витрати. Впровадження кластера, НА та централізованого керування зазвичай зменшує час на рутинні операції, оскільки адміністратор виконує типові дії один раз на рівні Datacenter, а також має можливість виконувати планові роботи без простою сервісів через міграції. Річний фонд часу на адміністрування до і після впровадження позначимо як $\boldsymbol{H_0}$і $\boldsymbol{H_1}$(людино-годин/рік). Економія трудових витрат:

$$\boldsymbol{\Delta H = H_0 - H_1}.$$

Вартість трудових витрат із урахуванням накладних коефіцієнтів (податки, організаційні витрати) доцільно оцінювати:

$$\boldsymbol{C_l = H \cdot r \cdot k_{ov}},$$

де $\boldsymbol{r}$це погодинна ставка, грн/год, $\boldsymbol{k_{ov}}$це коефіцієнт накладних витрат.

Для прикладу прийнято, що до впровадження витрачалося близько 20 год/місяць, після впровадження 12 год/місяць. Тоді

$\boldsymbol{H_0 = 20 \cdot 12 = 240}$ год/рік, $\boldsymbol{H_1 = 12 \cdot 12 = 144}$ год/рік,

$\boldsymbol{\Delta H = 96}$ год/рік.

За умови $\boldsymbol{r = 300}$ грн/годта $\boldsymbol{k_{ov} = 1,22}$річна економія на трудових витратах:

$$\boldsymbol{\Delta C_l = \Delta H \cdot r \cdot k_{ov} = 96 \cdot 300 \cdot 1,22 = 35136}\ \text{грн}.$$

Отже, сумарний річний економічний ефект за двома основними статтями (електроенергія та трудові витрати) становить

$$\boldsymbol{E}_{\text{year}} = \boldsymbol{\Delta C_e + \Delta C_l = 33358,08 + 35136 = 68494,08}$$ грн/рік.

Вартість розроблення та впровадження запропонованого інструментарію формується переважно з витрат праці, а також із витрат на додаткові компоненти, потрібні для коректного функціонування кластера і моніторингу. Оскільки Proxmox VE та базові компоненти моніторингу можуть використовуватися у варіанті з відкритою ліцензією, ліцензійні платежі в розрахунку приймаються нульовими, а вартість продукту розглядається як вартість створеного комплексу конфігурацій, регламентів, шаблонів і налаштувань, які забезпечують кластеризацію, HA та контроль метрик.

Вартість робіт з розроблення і впровадження оцінюється:

$$\boldsymbol{C_{dev} = H_{dev} \cdot r \cdot k_{ov}},$$

де $\boldsymbol{H_{dev}}$це трудомісткість робіт розроблення, налаштування та тестування. Для прикладу прийнято $\boldsymbol{H_{dev} = 140}$ год, що охоплює підготовку стенда, розгортання кластера, підключення спільного сховища, налаштування HA, організацію моніторингу, тестові відмови та документування результатів. Тоді

$$\boldsymbol{C_{dev} = 140 \cdot 300 \cdot 1,22 = 51240}$$ грн.

Додаткові матеріальні витрати, пов’язані з мережею і комутацією, можна оцінити зведено як $\boldsymbol{C_{eq}}$. Для прикладу прийнято $\boldsymbol{C_{eq} = 23000}$ грн(мережеві адаптери, комутація та кабельна інфраструктура для сегмента службового трафіку). Тоді повна вартість впровадженого продукту:

$$\boldsymbol{C}_{\text{total}} = \boldsymbol{C_{dev} + C_{eq} = 51240 + 23000 = 74240}$$ грн.

Період окупності визначається як відношення одноразових витрат на впровадження до річного ефекту:

$$T_{pb} = \frac{C_{\text{total}}}{E_{\text{year}}} = \frac{74240}{68494,08} \approx 1,08 \text{ року.}$$

Показник рентабельності інвестицій за рік (ROI) у спрощеному вигляді:

$$ROI = \frac{E_{\text{year}}}{C_{\text{total}}} \cdot 100\% \approx \frac{68494,08}{74240} \cdot 100\% \approx 92,3\%.$$

Отримані значення свідчать, що навіть за консервативних припущень окупність є близькою до одного року, а економічний ефект формується одночасно через енергозбереження і через скорочення трудовитрат на експлуатацію. Додатково в реальних умовах часто виникає непрямий ефект, який важко точно монетизувати в рамках короткого розрахунку, але він має практичну вагу. Йдеться про зменшення втрат від простоїв, зниження ризиків при планових роботах, а також можливість відкласти закупівлю нового обладнання завдяки консолідації, що збільшує корисний коефіцієнт використання вже наявних серверів.

Проведений розрахунок показників ефективності підтверджує доцільність впровадження запропонованого інструментарію. Енергетичний ефект проявляється у зниженні річного споживання електроенергії приблизно на 21,25 відсотка для прийнятого сценарію, що прямо трансформується в зменшення витрат на електроенергію. Економічний ефект доповнюється скороченням трудомісткості експлуатації за рахунок централізованого керування та автоматизованого відновлення сервісів. Сукупний річний ефект перевищує шістдесят вісім тисяч гривень, а орієнтовний строк окупності становить близько 1,08 року, що є прийнятним показником для інфраструктурних проєктів такого класу.

ЗАГАЛЬНІ ВИСНОВКИ

У кваліфікаційній роботі розглянуто та обґрунтовано підхід до оптимізації роботи дата-центрів шляхом віртуалізації обчислювальних ресурсів і побудови кластерного середовища. Показано, що перехід від ізольованих фізичних серверів до віртуалізованої інфраструктури створює передумови для ефективнішого використання CPU, оперативної пам’яті та дискових ресурсів, підвищує керованість сервісів і спрощує масштабування інфраструктури відповідно до зміни навантаження.

У ході дослідження проаналізовано сучасні платформи віртуалізації та обґрунтовано вибір інструментарію для практичної реалізації. Підтверджено доцільність використання кластерного підходу на базі Proxmox VE як рішення, що поєднує гіпервізор KVM, підтримку контейнеризації, механізми централізованого керування, міграції та інтеграції зі спільними сховищами, що є важливим для побудови відмовостійкої інфраструктури.

Практична частина роботи включала розгортання кластера Proxmox VE та деталізоване відтворення етапів його створення і приєднання вузлів. Отримано працездатну конфігурацію з коректним кворумом, що забезпечує узгоджене прийняття рішень у кластері та є базою для виконання операцій керування ресурсами. Підключено спільне сховище, що дозволило винести дискові дані віртуальних середовищ за межі локальних накопичувачів окремого сервера та створити технічну основу для міграцій і автоматичного відновлення сервісів.

Окремо реалізовано та перевірено механізми High Availability для керованих ресурсів. Експериментально підтверджено, що при відмові вузла кластер здатний автоматично переназначати ресурс і відновлювати його запуск на доступній ноді без ручного втручання, що зменшує простої та підвищує надійність роботи сервісів. Також встановлено практичні обмеження, пов’язані з втратою кворуму, які необхідно враховувати під час проектування мінімальної кількості вузлів та організації резервування.

Виконано оцінювання ефективності використання ресурсів у різних режимах експлуатації кластера. Показано, що балансоване розміщення навантаження забезпечує більш стабільний профіль ресурсоспоживання і створює запас продуктивності для пікових ситуацій, тоді як консолідація підвищує утилізацію активних вузлів і відкриває можливості енергозбереження та оптимізації витрат на обслуговування. Досліджено вплив живої міграції на інфраструктурні метрики, зокрема на затримки дискових операцій і прикладні показники якості сервісу, що дозволяє формувати обґрунтовані політики керування міграціями з урахуванням контрольних порогів.

У п’ятому розділі визначено ефективність впровадження запропонованого інструментарію за економічними й енергетичними показниками, а також оцінено трудомісткість розроблення та експлуатації. Розрахунки показали, що впровадження кластерної віртуалізованої інфраструктури забезпечує відчутний річний економічний ефект за рахунок зниження енергоспоживання та скорочення трудових витрат на адміністрування, а строк окупності рішення для прийнятого сценарію є близьким до одного року, що підтверджує практичну доцільність запропонованого підходу.

Таким чином, мету роботи досягнуто, а поставлені завдання виконано: здійснено аналіз предметної області, обрано та обґрунтовано інструментарій, реалізовано кластерне віртуалізоване середовище з механізмами відмовостійкості та спільним сховищем, проведено оцінювання результатів і визначено ефективність впровадження. Отримані результати можуть бути використані як методична й практична основа для проєктування та модернізації інфраструктури малих і середніх дата-центрів, а також для впровадження віртуалізації як інструменту оптимізації ресурсів і підвищення надійності сервісів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Mell P., Grance T. The NIST Definition of Cloud Computing (SP 800-145) : [Electronic resource]. National Institute of Standards and Technology, 2011. Режим доступу: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf (дата звернення: 17.11.2025).

2. Broadcom. VMware vSphere 8.0 Documentation (TechDocs) : [Electronic resource]. Режим доступу: https://techdocs.broadcom.com/us/en/vmware-cis/vsphere/vsphere/8-0.html (дата звернення: 17.11.2025).

3. Cocaña-Fernández A., Rodríguez-Soares J., Sánchez L., Ranilla J. Improving the energy efficiency of virtual data centers in an IT service provider through proactive fuzzy rules-based multicriteria decision making. *The Journal of Supercomputing*. 2019. Vol. 75. P. 1078–1093. DOI: 10.1007/s11227-018-2301-1. Режим доступу: https://link.springer.com/article/10.1007/s11227-018-2301-1 (дата звернення: 17.11.2025).

4. Kushwaha K. Layman guide to Platform Virtualization : [Electronic resource]. 2015. Режим доступу: https://kunalkushwaha.github.io/post/layman-guide-to-platform-virtualization/ (дата звернення: 17.11.2025).

5. Proxmox Server Solutions GmbH. Proxmox VE Administration Guide : [Electronic resource]. Режим доступу: https://pve.proxmox.com/pve-docs/pve-admin-guide.html (дата звернення: 17.11.2025).

6. NetApp. Containers vs. Virtual Machines (VMs) : [Electronic resource]. NetApp Blog, 2018. Режим доступу: https://www.netapp.com/blog/containers-vs-vms/ (дата звернення: 17.11.2025).

7. Docker Inc. Get started : [Electronic resource]. Docker Documentation. Режим доступу: https://docs.docker.com/get-started/ (дата звернення: 27.11.2025).

8. Google. GKE overview : [Electronic resource]. Google Cloud Documentation. Режим доступу: https://docs.cloud.google.com/kubernetes-engine/docs/concepts/kubernetes-engine-overview (дата звернення: 17.11.2025).

9. Кластеризація в Proxmox VE : [Electronic resource]. ProHoster. Режим доступу: https://prohoster.info/uk/blog/administrirovanie/klasterizacziya-v-proxmox-ve (дата звернення: 17.11.2025).

10. Proxmox Server Solutions GmbH. Ports : [Electronic resource]. Proxmox VE Wiki, 2022. Режим доступу: https://pve.proxmox.com/wiki/Ports (дата звернення: 17.11.2025).

11. The Green Grid. Power Usage Effectiveness (PUE) : [Electronic resource]. Режим доступу: https://www.thegreengrid.org/node/372 (дата звернення: 17.11.2025).

12. Proxmox Server Solutions GmbH. High Availability : [Electronic resource]. Proxmox VE Documentation. Режим доступу: https://pve.proxmox.com/pve-docs/chapter-ha-manager.html (дата звернення: 17.11.2025).

13. Proxmox Server Solutions GmbH. Storage: NFS : [Electronic resource]. Proxmox VE Wiki. Режим доступу: https://pve.proxmox.com/wiki/Storage%3A_NFS (дата звернення: 17.11.2025).

14. Proxmox Server Solutions GmbH. Cluster Manager : [Electronic resource]. Proxmox VE Documentation. Режим доступу: https://pve.proxmox.com/pve-docs/chapter-pvecm.html (дата звернення: 17.11.2025).

15. Proxmox Server Solutions GmbH. External Metric Server : [Electronic resource]. Proxmox VE Wiki. Режим доступу: https://pve.proxmox.com/wiki/External_Metric_Server (дата звернення: 17.11.2025).

16. Prometheus Authors. Monitoring Linux host metrics with the Node Exporter : [Electronic resource]. Prometheus Documentation. Режим доступу: https://prometheus.io/docs/guides/node-exporter/ (дата звернення: 17.11.2025).

17. prometheus-pve. prometheus-pve-exporter (Prometheus Proxmox VE Exporter) : [Electronic resource]. GitHub. Режим доступу: https://github.com/prometheus-pve/prometheus-pve-exporter (дата звернення: 17.11.2025).

18. Proxmox Server Solutions GmbH. Proxmox VE API : [Electronic resource]. Proxmox VE Wiki, 2025. Режим доступу: https://pve.proxmox.com/wiki/Proxmox_VE_API (дата звернення: 17.11.2025).

19. Пістун І. П., Тимочко В.О., Городецький І. М.. Березовецький А. П. Охорона праці (гігієна праці та виробнича санітарія): навч. посібн. / за ред. І.П. Пістуна. Львів: Тріада плюс, 2011. 224 с.

20. Охорона праці: практикум /І.П. Пістун, Ю.В. Кіт, А.П.Березовецький – Суми: Університетська книга, 2000. –205с.