КВАЛІФІКАЦІЙНА РОБОТА

Група МІПЗс-22

Куруляк О.О.

2024

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

на правах рукопису

Куруляк Олексій Олександрович

УДК 004.4

Оптимізація моделей та засобів реалізації хмарних розподілених систем репозиторіїв цифрових даних

Спеціальність 121 – «Інженерія програмного забезпечення»

Кваліфікаційна робота на здобуття кваліфікації магістра

Нормоконтроль

_____Стисло О.В.

(підпис, дата, розшифрування підпису)

Студент

_____ Куруляк О.О.

(підпис, дата, розшифрування підпису)

Допускається до захисту

Завідувач кафедри

_____к.т.н., доц. Ващишак С.П.

(підпис, дата, розшифрування підпису)

Керівник роботи

_____ к.т.н., доц. Демчина М.М.

(підпис, дата, розшифрування підпису)

Івано-Франківськ – 2024

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

Освітній ступінь: «магістр»

Спеціальність: 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____

« 19 » лютого 2024 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Куруляку Олексію Олександровичу

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи

Оптимізація моделей та засобів реалізації хмарних розподілених систем репозиторіїв цифрових даних

керівник роботи:

Демчина Микола Миколайович, кандидат технічних наук, доцент

затверджена наказом вищого навчального закладу від « 26 » червня 2023 року № 32/1 с

2. Термін подання студентом роботи 16.02.2024

3. Вихідні дані роботи: Формальні моделі, методи та алгоритми.

4. Зміст кваліфікаційної роботи (перелік питань, які потрібно розробити)

1. Аналіз рішень в області хмарних репозиторіїв цифрових даних
2. Дослідження методів обробки даних в розподілених системах.
3. Семантичне представлення та інтеграція цифрових даних.
4. Розробка семантичної моделі та архітектури хмарних цифрових рішень.

5. Дата видачі завдання 29.06.2023

КОНСУЛЬТАНТИ РОЗДІЛІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Розділ	Консультант (прізвище, ініціали та посада)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Аналіз сервісів та рішень в області хмарних репозиторіїв цифрових даних	26.09.2023	Виконано
2.	Дослідження методів та підходів обробки даних в розподілених системах	20.10.2023	Виконано
3.	Розробка семантичної моделі та архітектури хмарних цифрових рішень	15.11.2023	Виконано
4.	Семантичне представлення та інтеграція цифрових даних	30.11.2023	Виконано
5.	Формування висновків	09.12.2023	Виконано
6.	Оформлення пояснювальної записки	22.12.2023	Виконано
7.	Оформлення графічного матеріалу та підготовка до захисту роботи	11.01.2024	Виконано

Студент _____ Куруляк О.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Демчина М.М.
(підпис) (прізвище та ініціали)

Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Сторінка	Опис графічного матеріалу	Сторінка	Опис графічного матеріалу
16	Типове мережеве, мобільне та хмарне обчислювальне середовище	58	Архітектура комп'ютерної системи та програмно-апаратні інтерфейси, які може реалізувати гіпервізор
18	Фактори, які враховуються під час проведення цифрового аналізу даних у мобільних і хмарних середовищах	61	Стек семантичного вебу / багатошарова модель
24	Етапи методу Design Science	64	Приклад RDF концепцій суб'єкта, предиката та об'єкта

29	Пріоритет отримання доказів на основі порядку мінливості	71	Взаємодія між 3 рівнями архітектури пропонованого агрегатора інформації
30	Архітектура розподіленої системи	72	Чотирьохрівнева модель архітектури пропонованого агрегатора інформації
33	Загальна структура однорангової мережі (P2P)	77	Процес збору даних та реагування на дії
43	Архітектура розподіленої СУБД	79	Метод семантичного представлення та інтеграції даних
46	Основні типи NoSQL баз даних	81	Онтологічна модель концепцій захоплення мережевих пакетів
52	Взаємозв’язок контролю та відповідальності між Cloud постачальником послуг і споживачем	82	Онтологічна модель концепцій файлової системи
55	Ключові сутності в типовій моделі хмарної інфраструктури		

АНОТАЦІЯ

Кваліфікаційна робота присвячена оптимізації моделей та засобів реалізації хмарних розподілених систем репозиторіїв цифрових даних шляхом вдосконалення моделей послуг та сутностей, що надаються хмарними інфраструктурами для процесів віртуалізації та гіпервізії в хмарних інфраструктурах.

В першому розділі проаналізовано сервіси та рішення в області побудови розподілених хмарних репозиторіїв цифрових даних, описані технології в області хмарних рішень з точки зору безпеки. Приведені фактори та ознаки хмарних репозиторіїв цифрових даних, підходи і методи дослідження сервісів обробки та аналізу цифрових даних .

В другому розділі виконано дослідження методів та підходів обробки даних в розподілених системах та хмарних інфраструктурах, проведено опис методик отримання інформації з цифрових джерел та розподілених систем як вид хмарних цифрових рішень. Досліджені мобільні та хмарні середовища як вид розподілених систем, наведені моделі послуг та сутності, що надаються хмарними розподіленими інфраструктурами, представлені моделі хмарної інфраструктури. Наведені процеси віртуалізації та гіпервізори в хмарних розподілених інфраструктурах

В третьому розділі виконана розробка семантичної моделі та архітектури хмарних рішень репозиторіїв цифрових даних з використання семантичних веб-технологій для проектування. Виконана розробка вимог та стандартів до розподіленої системи репозиторіїв цифрових даних та спроектована архітектура масштабованої однорангової розподіленої системи. Також розроблена структура хмарної системи на основі архітектури розподілу типу P2P-da.

КЛЮЧОВІ СЛОВА: СЕМАНТИЧНА МЕРЕЖА, БЕЗПЕКА ДАНИХ, ХМАРНІ СЕРВІСИ, РОЗПОДІЛЕНА СИСТЕМА, ЦИФРОВЕ РІШЕННЯ, ВІРТУАЛІЗАЦІЯ, ХМАРНА АРХІТЕКТУРА.

SUMMARY

The qualification work is devoted to the optimization of models and means of implementation of cloud distributed systems of digital data repositories by improving models of services and entities provided by cloud infrastructures for virtualization and hypervision processes in cloud infrastructures.

The first section analyzes services and solutions in the field of building distributed cloud repositories of digital data, describes technologies in the field of cloud solutions from the point of view of security. Factors and characteristics of cloud repositories of digital data, approaches and research methods of digital data processing and analysis services are given.

In the second chapter, a study of data processing methods and approaches in distributed systems and cloud infrastructures was carried out, a description of the methods of obtaining information from digital sources and distributed systems as a type of cloud digital solutions was carried out. Mobile and cloud environments as a type of distributed systems are studied, service models and entities provided by cloud distributed infrastructures are presented, and cloud infrastructure models are presented. Virtualization processes and hypervisors in cloud distributed infrastructures are presented

In the third section, the development of the semantic model and the architecture of cloud solutions of digital data repositories using semantic web technologies for design is carried out. The development of requirements and standards for the distributed system of digital data repositories was carried out and the architecture of a scalable peer-to-peer distributed system was designed. A cloud system structure based on P2P-da type distribution architecture is also developed.

KEY WORDS: SEMANTIC WEB, DATA SECURITY, CLOUD SERVICES, DISTRIBUTED SYSTEM, DIGITAL SOLUTION, VIRTUALIZATION, CLOUD ARCHITECTURE.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

APT - Advanced Persistent Threat

BYOD - Bring Your Own Device

CBB - Cloud-based Backend

DHT - Distributed Hash Table

DBMS - Database Management System

GPS - Global Positioning System

HbH - Host-based Hypervisor

IoT - Internet of Things

IaaS - Infrastructure as a Service

KVM - Kernel Virtual Machine

MDM - Mobile Device Management

NIST - National Institute of Standards and Technology

OWL - Web Ontology Language

P2P - Peer to Peer

P2P-da - Peer to Peer Distribution Architecture

PaaS - Platform as a Service

RDF - Resource Description Framework

RIF - Rule Interchange Format

SaaS – Software as a Service

SPARQL - Simple Protocol and RDF Query Language

VMM - Virtual Machine Monitor

ВСТУП

Актуальність теми дослідження. Швидко зростаюче покоління Інтернет-сервісів, таких як електронна пошта, блоги, соціальні мережі, пошук та електронна комерція суттєво переосмислили поведінку та тенденції розвитку веб-користувачів, коли йдеться про створення, спілкування, доступ до вмісту, обмін інформацією та придбання продуктів. Професіонали в галузі стають свідками збільшення обсягу даних, що генеруються та споживаються через зростання кількості цих систем; це постійно зростаюча потреба в масштабованості та нові вимоги до додатків, нові виклики для традиційних систем управління реляційними базами даних (RDBMS).

В даний час, очевидною метою виробників систем і інструментів є полегшення завдання реалізації кожної програми як розподіленої, масштабованої та широкомасштабної, щодо доступних служб в Інтернеті (наприклад, служби Facebook, Flickr, YouTube, Zoho та LinkedIn). Технологія хмарних обчислень є відносно новою моделлю для розміщення програмного забезпечення. Хмарна модель спрощує трудомісткі процеси надання апаратного забезпечення, придбання апаратного забезпечення та розгортання програмного забезпечення; тому воно революціонізує спосіб комерціалізації обчислювальних ресурсів і послуг і доставки їх клієнтам. Зокрема, зміщується розташування цієї інфраструктури до мережі, щоб зменшити витрати, пов’язані з керуванням апаратними та програмними ресурсами.

Це означає, що хмара символізує те, що давно утримується уявлення про обчислення як корисну функцію, концепція економії масштабу. Такі принципи допомагають ефективно знизити вартість обчислювальної інфраструктури. На практиці, хмарні обчислення обіцяють низку переваг для розгортання програмних додатків, такі як модель оплати за використання, короткий час виходу на ринок, сприйняття (практично) необмежених ресурсів і нескінченної масштабованості.

Розвиток хмарних технологій був дещо руйнівним. Переваги моделі хмарних обчислень відкривають нові шляхи для розгортання нових програм які були економічно недоцільними в умовах традиційної інфраструктури. Тому хмара стає все більш популярною платформою для розміщення програмного забезпечення в різних областях, таких як роздрібна торгівля, фінанси, новини, соціальні мережі. Зростання кількості додатків також приносить користь через величезне збільшення масштабу даних, які генеруються та споживаються. Ось чому система баз даних, розміщена в хмарі, що забезпечує їх живлення є критично важливим компонентом програмного стеку новітніх додатків.

Мета і завдання дослідження. **Метою кваліфікаційної роботи** є дослідження методів та підходів обробки даних в розподілених системах та хмарних інфраструктурах, розробці методик отримання інформації з цифрових джерел для побудови оптимальних розподілених репозиторіїв даних як виду хмарних цифрових рішень.

Для досягнення поставленої мети необхідно розв’язати такі завдання:

- виконати аналіз сервісів та рішень в області хмарних репозиторіїв цифрових даних;
- розробити методику отримання інформації з цифрових джерел;
- провести дослідження методів та підходів обробки даних в розподілених системах та хмарних інфраструктурах;
- дослідити моделі послуг та сутності, що надаються хмарними розподіленими інфраструктурами;
- виконати розробку семантичної моделі та архітектури хмарних цифрових рішень.

Об’єкт дослідження. є самі мобільні та хмарні розподілені середовища, моделі послуг та сутності, що надаються хмарними інфраструктурами для процесів віртуалізації та гіпервізії.

Предметом дослідження є оптимізація моделей та засобів реалізації хмарних розподілених систем репозиторіїв цифрових даних.

Методи дослідження базуються на використанні методів семантичного аналізу, методів побудови семантичних мереж та масштабування розподілених ресурсів, методів та моделі P2P, методів системного аналізу, моделі та сутності багаторангових мереж та кластерів даних.

Наукова новизна одержаних результатів полягає у тому, що на основі ґрунтовного аналізу побудови хмарних систем запропоновано використання гіпервізора в компоненті архітектури Host-based Hypervisor (HbH), що усуває неоднорідність різноманітних програмних платформ та дозволяє створити єдину точку збору цифрових даних. В даному контексті гіпервізор сприяє довірі до процесу отримання даних і цілісності джерел даних, оскільки забезпечує більш привілейовану платформу, з якої можна отримати цифрові дані.

Практичне значення одержаних результатів полягає в створенні концепції побудови сервісів та рішень в області хмарних репозиторіїв цифрових даних на основі аналізу технологій в області хмарних рішень з точки зору безпеки та визначення факторів та ознак хмарних технологій зберігання та обробки цифрових даних.

Апробація результатів дослідження. Матеріали дослідження було представлено у матеріалах І Всеукраїнської науково-практичної інтернет конференції “ІТ екосистема: цифровізація бізнес-процесів в умовах війни”, у тезах доповіді “Підтримка прийняття рішень у хмарних додатках”.

Структура. Кількість розділів – 3. Загальний обсяг основної частини – 88 сторінок. Список використаних джерел містить – 45 позицій.

РОЗДІЛ 1. АНАЛІЗ СЕРВІСІВ ТА РІШЕНЬ В ОБЛАСТІ ПОБУДОВИ РОЗПОДІЛЕНИХ ХМАРНИХ РЕПОЗИТОРІЇВ ЦИФРОВИХ ДАНИХ

1.1 Аналіз технологій в області хмарних рішень з точки зору безпеки

Обчислювальні пристрої та мережі поступово проникли майже в усі форми людської діяльності. Повсякденна людська діяльність, така як покупки, банківська справа, спілкування, транспорт, звернення до державних послуг і навіть медичних послуг, увібрала деякі аспекти обчислювальних систем [1]. Приклади їх реалізації включають: платформи електронної комерції, портали електронного банкінгу, рішення для мобільних платежів, платформи соціальних медіа, системи відстеження GPS, портали електронного уряду та електронні системи охорони здоров’я. Пов’язані та комп’ютеризовані форми цієї автоматизованої чи комп’ютеризованої людської діяльності надали учасникам ряд переваг, головним чином з точки зору простоти використання, зручності та зменшення паперової роботи.

Отже, це призвело до створення великої кількості оцифрованих даних, які зберігаються на обчислювальних пристроях, а також передаються через мережі даних до різних кінцевих точок. Часто ці пристрої та мережі беруть участь у транзакціях, які включають досить конфіденційні особисті або корпоративні дані, які вимагають безпеки даних під час зберігання, обробки або передачі через ці пристрої та мережі. Зазвичай це забезпечується впровадженням механізмів безпеки, які забезпечують конфіденційність, цілісність, доступність, автентифікацію, контроль доступу та звітність [2].

Криптографічні схеми, реплікація та резервування, керування ідентифікацією, реєстрація подій, документація політики, сканування проти зловмисного програмного забезпечення та виявлення вторгнень є поширеними формами впровадження цих механізмів безпеки. Однак

загальноприйнятим фактом у спільноті безпеки є те, що механізми безпеки ніколи не бувають повністю безпомилковими і що вони врешті-решт виходять з ладу. В [3] зазначається, що «будь-яка мережева організація, ймовірно, зазнає спорадичного або постійного компромісу». Саме з цієї причини потрібні більш реактивні форми механізмів безпеки, такі як цифровий моніторинг безпеки мережі, щоб доповнити вже існуючі механізми безпеки, орієнтовані на попередження, запобігання, виявлення, пом'якшення або відновлення. Такі реактивні механізми збільшують загальну безпеку суб'єкта, доповнюючи механізми безпеки, коли виникають збої, що дозволяє їм зрештою виявляти порушення безпеки та вчитися на них, навіть якщо це може бути постфактум.

Крім того, відбувся великий прогрес в інструментах і техніках, які використовуються для здійснення зловмисних дій. Використання криптографії, однорангових технологій, алгоритмів генерації доменів [4], методів швидкого потоку, спеціальних пакувальників. приховані канали та інші методи ухилення від аналізу зловмисного програмного забезпечення [5] допомогли зловмисній діяльності стати більш витонченою, що ускладнило її виявлення, стримування або запобігання. Це зробило стандартні механізми запобігання та виявлення, такі як брандмауери, системи виявлення вторгнень, фільтри спаму та програми захисту від зловмисного програмного забезпечення, досить неефективними для забезпечення безпеки.

Цифровий аналіз даних визначається як «використання науково розроблених і перевірених методів для збереження, збору, перевірки, ідентифікації, аналізу, інтерпретації, документування та представлення цифрових доказів, отриманих із цифрових джерел, з метою полегшення або подальшої реконструкції подій, визнаних злочинними, або допомоги у передбаченні несанкціонованих дій, які можуть порушити заплановані операції». Процедуру фактичних кроків, вжитих під час виконання цифрового аналізу було розбито на низку різних моделей процесу цифрової підготовки

даних. Серед них спостерігаються невеликі варіації через найменування та абстракцію процесу.

В даному дослідженні ми узагальнюємо моделі процесу цифрового дослідження на наступні 4 фази:

- підготовка та виявлення доказів;
- збирання (придбання) та збереження доказів;
- аналіз доказів та документування висновків;
- презентація висновків, усунення або відновленняю.

Незважаючи на те, що ці моделі процесів здаються послідовними, на практиці досить поширеним є застосування ітераційного підходу, коли це необхідно вздовж потоку процесу. Тобто, впродовж звичайно послідовного процесу, може бути визнано необхідним виконати попередні кроки ще раз, щоб зміцнити та покращити розуміння конкретної справи. Документування всієї процедури в цілому виконується протягом усього процесу, однак це більш інтенсивно необхідно на етапі аналізу.

Останнім часом використання нестандартних обчислювальних пристроїв у вигляді інфраструктури хмарних обчислень і мобільних пристроїв, зросло у своїй популярності і тепер вони починають займати важливе місце в суспільстві.

Крім того, хмарні обчислення та середовища мобільних пристроїв демонструють деякі параметри для цифрового дослідження, як зазначено нижче:

- Вони набули широкого поширення як серед корпоративних, так і для особистих користувачів для різноманітних додатків, таким чином генеруючи, передаючи та зберігаючи величезні обсяги даних.
- Завдяки популярності цих технологій було розроблено багато додатків, які використовують різноманітні типи даних і протоколи зв’язку, що ускладнює проведення аналізу.
- Багато даних, що передаються через хмарні середовища та мобільні пристрої, вважаються досить конфіденційними, наприклад

електронні листи, календарі, зображення, відео, SMS, розмови в соціальних мережах, дані про місцезнаходження, біометричні дані, ключі, паролі, фінансові дані, плани компанії, дані клієнтів або відомості про співробітника. Це робить їх основними цілями для невиправданих зловмисних дій.

- Приватні пристрої користувачів використовують загальнодоступні ресурси (обчислювальні пристрої чи мережеву інфраструктуру), над якими вони не мають повного однозначного контролю, а тому не можуть повністю довіряти.
- Окремі пристрої, як правило, обмежені в ресурсах і спеціалізуються з точки зору їх функціональності та можливостей. Таким чином, здійснення моніторингу безпеки або діяльності з цифрової криміналістики може бути обмежено та виходити за межі їхньої звичайної діяльності.
- Як наслідок їхньої повсюдності, вони є динамічно розподіленими ресурсами (сховищем, обчислювальними або мережевими ресурсами), які можуть бути відкликані, перерозподілені або ненавмисно втрачені. Таким чином доступність пристроїв і, отже, їхніх властивих джерел доказів може бути втрачено.

На рисунку 1.1 показано зображення мережевого середовища мобільних і хмарних обчислень.

Рисунок 1.1 – Типове мережеве, мобільне та хмарне обчислювальне середовище

Підсумовуючи, ці характеристики забезпечують цікавий набір параметрів для проведення цифрових розслідувань, оскільки вони втілюють кілька проблем, які раніше зустрічалися в окремих випадках у традиційних методах цифрового аналізу, однак тепер усі вони зібрані та об’єднані в єдине мережеве середовище, яке почалося займати важливе місце під час цифрової обробки великого потоку даних.

1.2 Фактори та ознаки хмарних репозиторіїв цифрових даних

Цифрові розслідування щодо безпеки даних в основному ґрунтується на 2 факторах, а саме надійності та повноті, де перший є основною проблемою, тоді як другий стає все важче досягти, і тому поступово він відіграє меншу роль [5]. Обидва ці фактори вимагаються від цифрових доказів, а також від дій, що виконуються в рамках моделі процесу цифрового аналізу, тобто: підготовка та ідентифікація даних, отримання джерела даних, аналіз і звітність про висновки.

Надійність оцінюється через рівень цілісності отриманих цифрових даних, а також через надійність дій, які виконуються протягом цифрового аналізу.

Надійність вимагає, щоб дані та процедури, які з ними використовуються, підлягали науковій суворості з точки зору можливості фальсифікації, експертної перевірки, перевірки. перевірені раніше та мають доступний рівень помилок. Крім того, існує також вимога, щоб процедури були прийнятними серед спільноти експертів у галузі. Ці вимоги в цілому сприяють довірі до процедур.

Цілісність зазвичай підтверджується підтримкою хеш - значень даних доказів [6]. Це доповнюється документацією фактичних дій процесу цифрового аналізу даних з точки зору того, хто мав доступ до даних, де вони були та які процедури були застосовані до нього.

Повнота часто вважається складною (якщо не майже неможливою) вимогою для досягнення. Це головним чином пов’язано із зазвичай великими обсягами даних, які потенційно можуть використовуватися як докази. Типові випадки сьогодні можуть включати терабайти даних із різноманітних джерел, і поки що немає загальноприйнятих рішень. На даний момент ведуться дебати щодо того, чи потрібна повнота, чи сортування є шляхом вирішення цієї проблеми.

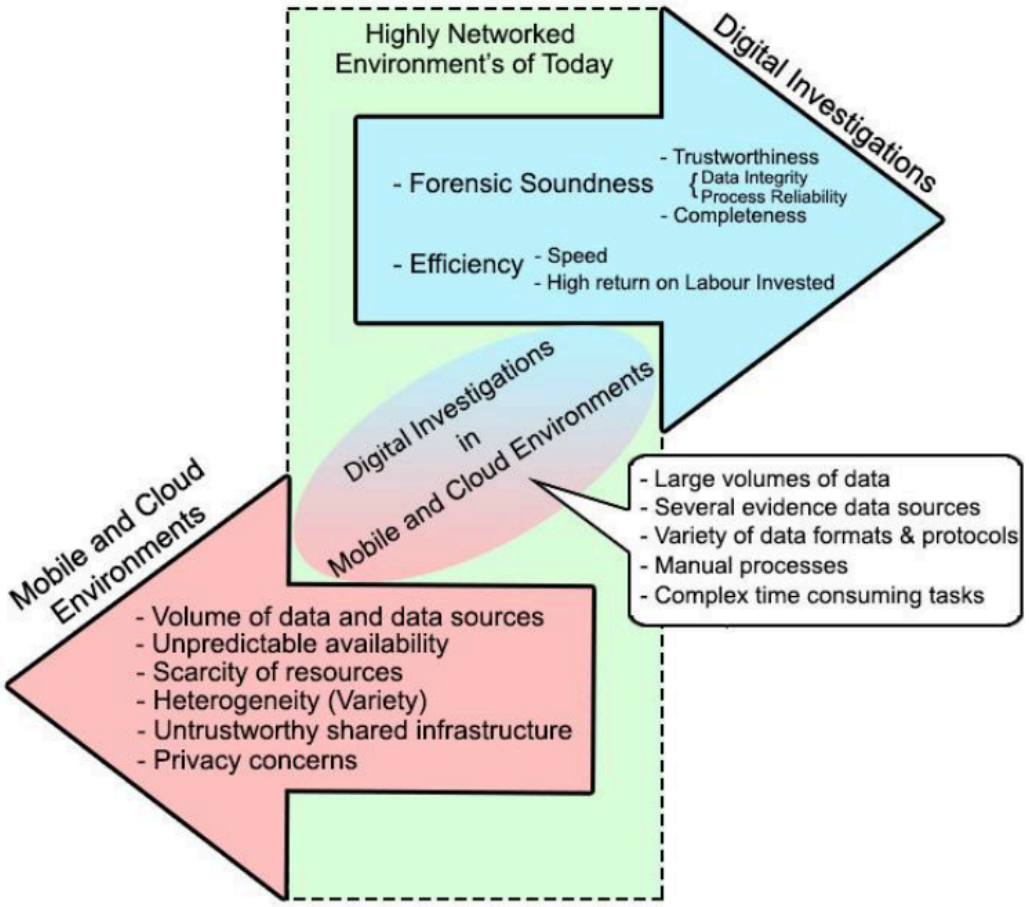


Рисунок 1.2 – Фактори, які враховуються під час проведення цифрового аналізу даних у мобільних і хмарних середовищах

●

Як і будь-яке інший цифровий аналіз даних, що проводиться в мобільних і хмарних середовищах, також вимагають обґрунтованості та швидкості їх виконання. Однак у цьому середовищі є певні фактори, які

ускладнюють виконання цих вимог у повному обсязі. Ці фактори описані нижче:

- Наявність великих обсягів даних: типові цифрові дослідження сьогодні включають кілька пристроїв і великі обсяги даних. Оскільки мобільні пристрої поступово стають все краще оснащеними ресурсами, створюється та зберігається більше даних. Хмарні інфраструктури зазвичай мають багато користувачів і великі обсяги даних.
- Доступність: переривчасте підключення до мережі або зміна фізичних мережевих ресурсів на льоту, як правило, спостерігається в спільній інфраструктурі мобільних і хмарних обчислювальних середовищ. Це означає, що доступність пристроїв може вплинути, коли вони перемикаються між мережевими ресурсами. Крім того, вони можуть бути припинені та перезапущені в іншому місці, тому будь-які дії, що відбуваються на них, повинні мати можливість безперешкодно перезапускатися або продовжуватися, незважаючи на перерви, які можуть виникнути. Недоліки в доступності впливають на доступність джерел, які є ключовою відправною точкою кожного цифрового аналізу даних.
- Дефіцит ресурсів: мобільні пристрої, вбудовані пристрої та хмарні пристрої, як правило, спрямовані на виконання дуже конкретних завдань і, таким чином, мають обмежені обчислювальні ресурси (тобто обсяг пам’яті, оперативна пам’ять і швидкість процесора) для виконання цих завдань. Це обмежує їх здатність виконувати інтенсивні обчислювальні процеси, типові для цифрової криміналістики та механізмів безпеки в цілому. Крім того, мобільні та вбудовані пристрої зазвичай мають обмежений заряд акумулятора, оскільки вони часто призначені для портативних пристроїв. Це може вплинути на їхню доступність, а також на здатність виконувати завдання, пов’язані з безпекою чи цифровою експертизою.
- Неоднорідність: обчислювальні пристрої загалом, але особливо мобільні та хмарні пристрої, мають велику кількість варіантів операційних систем, програмних додатків і доступних протоколів зв’язку. Крім того, з

розвитком технологій постійно випускаються нові версії, оскільки вони постійно змінюються. Це створює проблему для цифрової криміналістики, оскільки вона повинна мати можливість керувати такими високими рівнями різноманітності джерел даних і типів даних. Крім того, зворотне проектування, синтаксичний розбір та аналіз власних форматів даних може бути складним і складним процесом, особливо якщо документація щодо таких форматів є малою.

- Ручні або квазіавтоматизовані процеси цифрової експертизи: більшість етапів моделі процесу цифрової експертизи виконуються переважно вручну та вимагають втручання людини-оператора. Наприклад, на етапі отримання даних, значною мірою людина-оператор повинен визначити відповідні джерела для отримання, а також ініціювати фактичний процес отримання. Часто також необхідна фізична близькість до пристрою, щоб підключити спеціалізоване обладнання для блокування запису для виконання процесу збору даних. Фаза аналізу також залежить від людини-оператора, щоб зрозуміти та співвіднести докази, особливо якщо вони надходять із кількох джерел. Деякі завдання, такі як індексування даних для цілей пошуку, а також фільтрація відомих файлів є частиною комп’ютерних завдань, спрямованих на прискорення процесу. У мобільних і хмарних середовищах складно отримати й проаналізувати докази через мінливий, тимчасовий і швидкоплинний характер пристроїв. Таким чином, механізми цифрової криміналістики в таких середовищах повинні мати можливість швидко збирати докази без необхідності фізичної близькості, а також дозволяти швидкий аналіз за допомогою автоматизації. Це допоможе пришвидшити судово-медичний процес, а також увімкне подальше отримання, реагування або механізми відновлення якомога раніше.

- Внутрішня відсутність довіри/контролю: через спільні ресурси та інфраструктуру, на які покладаються мобільні та хмарні пристрої, існує вроджена відсутність довіри та контролю. Це тому, що постачальник послуг або інші користувачі, які використовують спільні ресурси, можуть мати різні,

суперечливі або навіть зловмисні наміри. Це створює проблему щодо того, як забезпечити довіру та контроль у такому середовищі, особливо під час проведення цифрового розслідування, щоб підтримувати цілісність даних і достовірність процесу.

- Конфіденційність: мобільні пристрої зазвичай зберігають багато конфіденційних даних (таких як геолокаційні дані, повідомлення, банківська інформація та біометричні дані), оскільки вони тісно пов’язані зі своїми користувачами завдяки багатій функціональності, яку вони надають. Хмарні засоби зберігання та обробки так само широко використовуються для зберігання конфіденційних даних окремих осіб (наприклад, фотографій, відео та документів), а також компаній (наприклад, інформації про співробітників, даних про здоров’я та інформації про клієнтів). Цифрові розслідування мають на меті отримати доступ, зібрати й проаналізувати якомога більше даних. У такому середовищі це може призвести до розкриття приватної інформації.

З наведених вище проблем можна побачити, що це особливо складне завдання — ефективне виконання цифрової експертизи в мобільному та хмарному середовищі. Деякі з цих проблем окремо впливають на різні етапи моделі процесу цифрового аналізу даних, а в поєднанні ефективно сповільнюють весь процес, сприяючи вже відомим і відчутним відставанням у цифрових розслідуваннях. Зокрема, проблеми, які безпосередньо спричиняють повільний процес цифрового дослідження, це обсяг даних, періодична доступність пристроїв, дефіцит ресурсів, неоднорідність пристроїв і даних і переважно ручні процеси попереднього виходу.

1.3 Підходи і методи дослідження сервісів обробки та аналізу цифрових даних

Згідно [8] існують дві основні дослідницькі парадигми, які характеризують дослідження в галузі інформаційних систем, тобто наука про поведінку та наука про дизайн. Стверджується, що парадигма поведінкової

науки «прагне розробити та перевірити теорії, які пояснюють або передбачають людську чи організаційну поведінку». З іншого боку, вони стверджують, що парадигма науки про дизайн «прагне розширити межі людських і організаційних можливостей шляхом створення нових та інноваційних артефактів».

Оскільки це дослідження не має на меті «пояснити чи передбачити поведінку людини чи організації», парадигма поведінкової науки вважається непридатною для цього дослідження. Парадигма Design Science здається більш прийнятною, оскільки це дослідження спрямоване на вирішення реальної проблеми неефективності обробки цифрових даних шляхом розробки артефакту, який сприяє більшій автоматизації дій у мобільних і хмарних середовищах. Він має на меті розширити межі поточних практик цифрових розслідувань у технологічний спосіб шляхом розробки архітектури та реалізації частин цієї архітектури, щоб забезпечити автоматизацію та, таким чином, сприяти більш ефективним цифровим розслідуванням. Оскільки Design Science «підтримує прагматичну дослідницьку парадигму, яка закликає до створення інноваційних артефактів для вирішення реальних проблем», цей підхід виглядає ідеальним для дослідження.

Метод Design Science використовується для керівництва дослідженнями, проведеними в цій роботі. Тут вони визначають науку про дизайн як «наукове дослідження та створення артефактів, які розробляються та використовуються людьми з метою вирішення практичних проблем загального інтересу». Артефакт визначають як «об’єкт, виготовлений людьми з наміром використовувати його для вирішення практичної проблеми», де практична проблема описується як «небажаний стан, або, точніше, розрив між поточним станом і поточним станом». бажаний стан, як його сприймають учасники практики» [12].

Вони також виділяють чотири типи артефактів, а саме: конструкції, моделі, методи та інстанції. Конструкції складаються з основоположних

термінів, позначень, визначень і понять, необхідних для опису області проблеми або можливе рішення.

Моделі являють собою об’єкти, зібрані разом, щоб зобразити конкретну проблему чи рішення. Вони допомагають проаналізувати проблеми або показують, чому вони можуть бути складними. Вони також допомагають описати потенційні рішення, які можуть допомогти у створенні інших артефактів, що вирішують практичну проблему, а також допомагають прогнозувати майбутню поведінку об’єктів або систем. Приклади моделей включають системні архітектури та онтології домену. Методи описують вказівки або процеси, як вирішити проблему. Приклади можуть включати алгоритми, «емпіричні правила» або найкращі практики. Екземпляри — це фактичні реалізації у формі робочих систем, які можуть вирішити певну проблему на практиці.

В даній роботі ми досліджуємо декілька артефактів, головним чином у формі моделей, методів та екземплярів. Розроблено та пізніше вдосконалено модель архітектури системи для проведення цифрових досліджень у мобільному та хмарному середовищах. Моделі також створюються у формі доменних онтологій, які використовуються для опису доказових артефактів з різних джерел.

Запропоновано метод інтеграції кількох різнорідних джерел доказів, щоб допомогти досягти кореляції та, зрештою, простого міркування на основі джерел доказів.

Розроблено екземпляри частини архітектури системи, зосередженої на отриманні цифрових доказів, а також екземпляр методу для інтеграції, кореляції та виконання простих міркувань на кількох різнорідних наборах даних доказів.

В [19] також описують загальний 5-етапний метод для виконання наукових досліджень дизайну. Ця структура зображена на рисунку 1.5 та є основою для виконання цього дослідження.

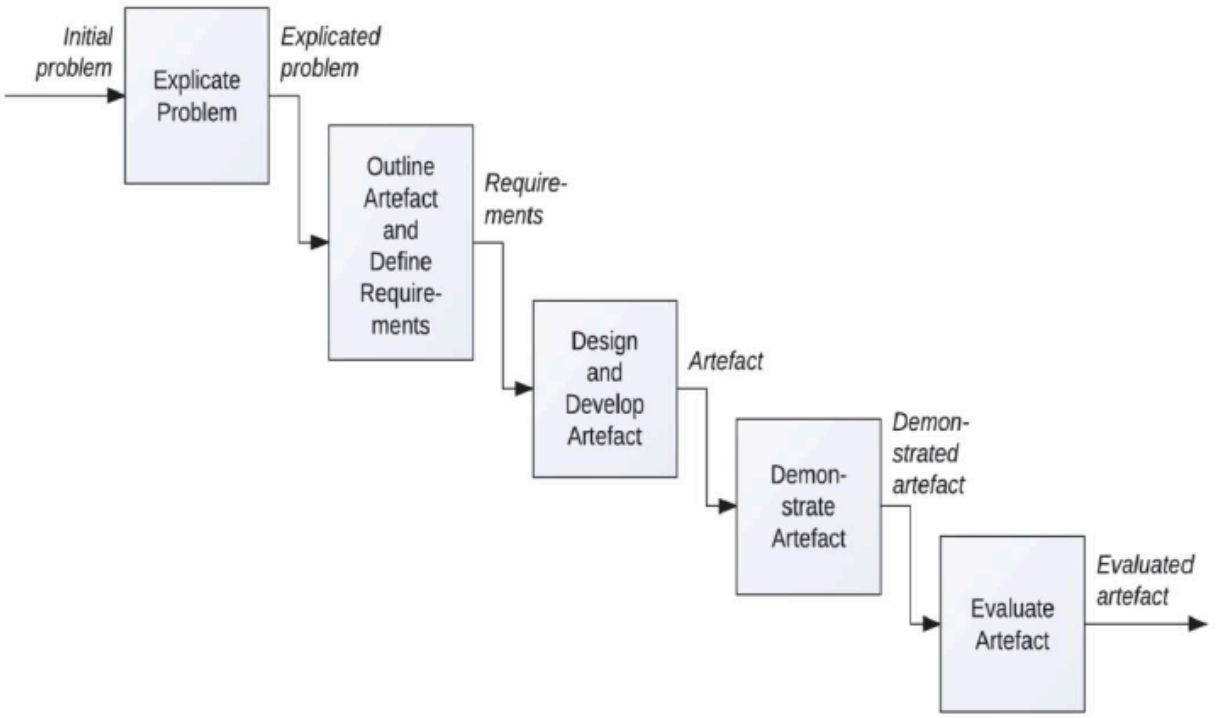


Рисунок 1.3 – Етапи методу Design Science

У минулому організації мали хороший огляд своєї інфраструктури та інвентаризації IT-активів, але це стає все складніше. BYOD (Bring Your Own Device) стає все популярнішим, співробітники все частіше використовують мобільні пристрої, більше інфраструктури передається аутсорсингу або працює у зовнішніх хмарних середовищах. Навіть поняття фізичної «системи» або «пристрою» змінюється, оскільки і сервери і робочі столи клієнтів стають віртуалізованими. Це впливає на цифрову безпеку, оскільки традиційні методи в деяких випадках більше не можуть проводитися таким же чином [8]. Поширеність мережевих технологій спричиняє зміну розташування цифрових даних. Те, що раніше здебільшого знаходилося на окремих дисках, пов’язаних з окремими особами, тепер розподіляється між віддаленими мережевими машинами, під контролем кількох організацій і розпорошено в кількох юрисдикціях.

Наведені вище аргументи передають суть поточного ландшафту цифрових даних. З ними поступово стає все важче боротися через збільшення

використання нестандартних пристроїв у вигляді мобільних пристроїв і віртуалізованих хмарних інфраструктур, якими керують із кількох різних місць. Сильно об’єднаний у мережу характер цих пристроїв підвищив їх повсюдність і ще більше сприяв поширенню цих технологій. З’явилися численні програми для цих технологій, що призвело до збільшення кількості даних, що генеруються, і різноманітності їх форматів. Таким чином, цифрові розслідування вимагають рішення, яке б допомогло впоратися з випадками, пов’язаними з такими середовищами.

У цій роботі пропонується автоматизація різних етапів процесу цифрового аналізу цифрових даних за допомогою 4-рівневої архітектури системи. Основна увага зосереджена на етапі збору даних і етапі аналізу даних моделі процесу цифрового аналізу.

Щоб досягти архітектури та аспектів автоматизації, які задіяні, необхідно коротко згадати та пояснити деякі базові знання. Це робиться з метою кращого розуміння архітектури, мотивації для поширення використовуваних технологій, а також для повноти.

Сфери, які необхідно коротко описати:

- Цифровий аналіз, включаючи концепцію цифрового місця знаходження даних, ідею цифрових доказів і стан інструментів цифрового аналізу.
- Концепція однорангових мереж (P2P), включаючи категорії накладень мереж P2P, а також переваги та недоліки, які вони привносять у мережевий зв’язок.
- Концепція семантичних веб-технологій, включаючи модель «Semantic Web layer cake», Resource Description Framework (RDF), мову веб-онтології (OWL) і простий протокол і мову запитів RDF (SPARQL).
- Мобільні та хмарні середовища, включаючи концепції, технології та протоколи, що забезпечують зв’язок навколо мобільних пристроїв; концепції та технології хмарних обчислень, включаючи віртуалізацію. Ці сфери описані в наступних розділах.

Висновки до розділу 1

В даному розділі представлено аналіз сервісів та рішень в області розподілених хмарних репозиторіїв цифрових даних, описана загальна сфера аналізу сервісів та рішень в області хмарних архівів цифрових даних, яка розглядається в цій роботі, а також визначено проблемний простір і мета дослідження. Також описуються підходи і методи дослідження сервісів обробки та аналізу цифрових даних.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ МЕТОДІВ ТА ПІДХОДІВ ОБРОБКИ ДАНИХ В РОЗПОДІЛЕНИХ СИСТЕМАХ ТА ХМАРНИХ ІНФРАСТРУКТУРАХ

2.1 Опис методик отримання інформації з цифрових джерел

Цифрове розслідування може бути призначено під час або після того, як сталася певна діяльність із застосуванням цифрових пристроїв або технологій. Метою цього є вивчення та реконструкція подій із цифрових джерел, щоб зрозуміти діяльність і запобігти виникненню небажаної діяльності в майбутньому. Виходячи з визначення цифрового розслідування є кілька важливих концепцій, на які слід звернути увагу:

- Перша ідея полягає у використанні «науково отриманих і перевірених методів».
- Наступна концепція — це ідея процесу, який використовується для полегшення цифрового аналізу, як це приблизно викладено як «збереження, збір, перевірка, ідентифікація, аналіз, інтерпретація, документування та збереження».
- Інша концепція – це концепція «цифрових доказів». Це, мабуть, найважливіша частина визначення в тому, що цифровий доказ — це те, до чого застосовується весь процес.

Цифрові докази отримані з цифрових джерел і наступний процес є важливими аспектами цифрового розслідування.

Цифрові докази були визначені різними авторами по-різному, однак усі вони намагаються охопити деякі з ключових аспектів такого артефакту. Деякі з цих визначень пропонуються нижче:

- «Будь-які дані, збережені або передані за допомогою комп’ютера, які підтверджують або спростовують теорію про те, як сталася певна дія» [13].

- «Інформація, що має доказову силу, зберігається або передається в двійковій формі» [14].
- «Будь-які цифрові дані, які містять надійну інформацію, яка підтверджує або спростовує гіпотезу про інцидент» [15].

З цих визначень ми можемо підсумувати, що цифрові докази включають будь-яку форму даних, які присутні на комп'ютері або передаються між комп'ютерами в цифровому форматі, які можуть допомогти підтвердити або спростувати теорію про певну діяльність, яка перевіряється.

Дані, які містять те, що називається цифровим доказом, походять від компонентів обчислювальної системи. Більш конкретно, загальні джерела доказів включають дані, знайдені в сховищі, обчислювальних або комунікаційних компонентах обчислювальної системи, яка включає дискове сховище, флеш-пам'ять, оперативну пам'ять, кеші та мережеві інтерфейси. Зазвичай вони надходять у формі образів дисків, записів пам'яті, записів мережевого трафіку та системних або мережевих журналів.

Стверджується, що «цифрові докази походять із безлічі джерел, включаючи жорсткі диски комп'ютерів і носії резервних копій, повідомлення електронної пошти в реальному часі, журнали чатів, записи провайдерів, веб-сторінки, цифровий мережевий трафік, локальні та віртуальні бази даних, цифрові довідники, бездротові пристрої, карти пам'яті та цифрові камери». Різноманітність і форми цифрових доказів та їх джерел постійно змінюються, оскільки технологічний прогрес змінює цифровий ландшафт.

Визначено 3 ключові властивості цифрових доказів, які впливають на спосіб проведення цифрових розслідувань. Ці 3 властивості:

1. Затримка: це відноситься до прихованої природи цифрово закодованих даних, оскільки набір двійкових даних у формі одиниць і нулів сам по собі не передає жодної інформації, однак під час аналізу можуть виникати різні інтерпретації залежно від мети інструменту аналізу.

2. Точність: Це стосується того факту, що цифрові дані, знайдені можна вільно копіювати та розглядати як оригінали, за умови, що процес копіювання може бути доведено точним.

3. Нестабільність: Це виникає через властивість вірності, оскільки цифрові дані можуть бути легко змінені навмисно або ненавмисно.

Крім того, виникла потреба підтвердити достовірність цифрових даних, щоб переконатися, що не було зроблено жодних несанкціонованих змін з моменту, коли вони були отримані. Це призвело до підтримки «ланцюжок зберігання», яка вимагає запису інформації про те, хто був залучений, де, коли та як цифрові докази були зібрані або перейшли в руки. Це також вимагає перевірки цілісності доказів за допомогою функцій хешування. У практиці це методи, які традиційно використовувалися для забезпечення обґрунтованості цифрових доказів, однак наразі їм важко впоратися з постійним прогресом технологій, який ускладнює отримання повних і немодифікованих копій цифрові докази в сучасних цифрових місцях злочинів, включаючи мобільні та хмарні середовища [17].

«Порядок мінливості» полягає в тому, що через тимчасовий характер цифрових даних пріоритет слід надавати збору певних джерел доказів перед іншими через їхню високу швидкознижуваність. На рисунку 2.1 зображено порядок мінливості та джерела доказів, які вона розглядає.

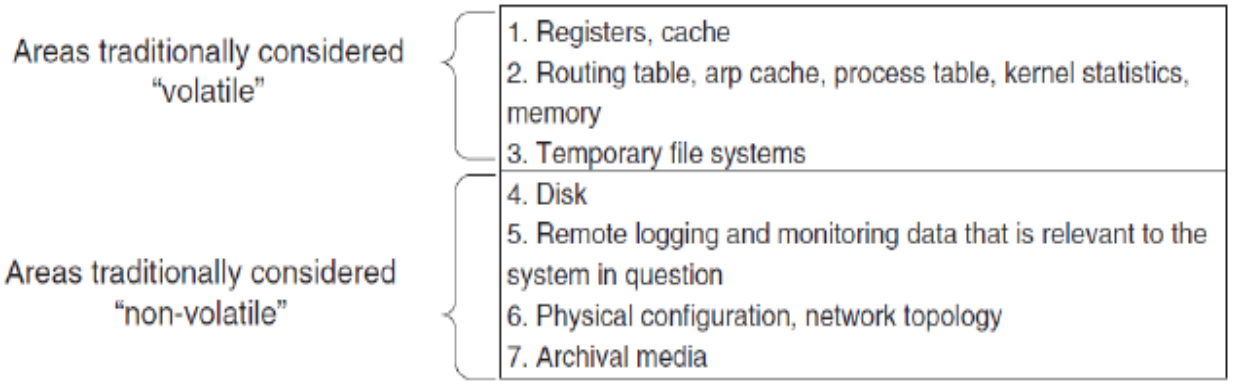


Рисунок 2.1 – Пріоритет отримання доказів на основі порядку мінливості

2.2 Розподілені системи як вид хмарних цифрових рішень

Розподілена система описується як система, в якій програмні та апаратні компоненти взаємодіють і координують усі свої дії виключно через передачу повідомлень через комп'ютерні мережі [18]. Основною метою побудови та використання розподілених систем є спільне використання ресурсів, як правило, з метою підвищення продуктивності за допомогою паралельної обробки та підвищення доступності у разі збоїв між компонентами.

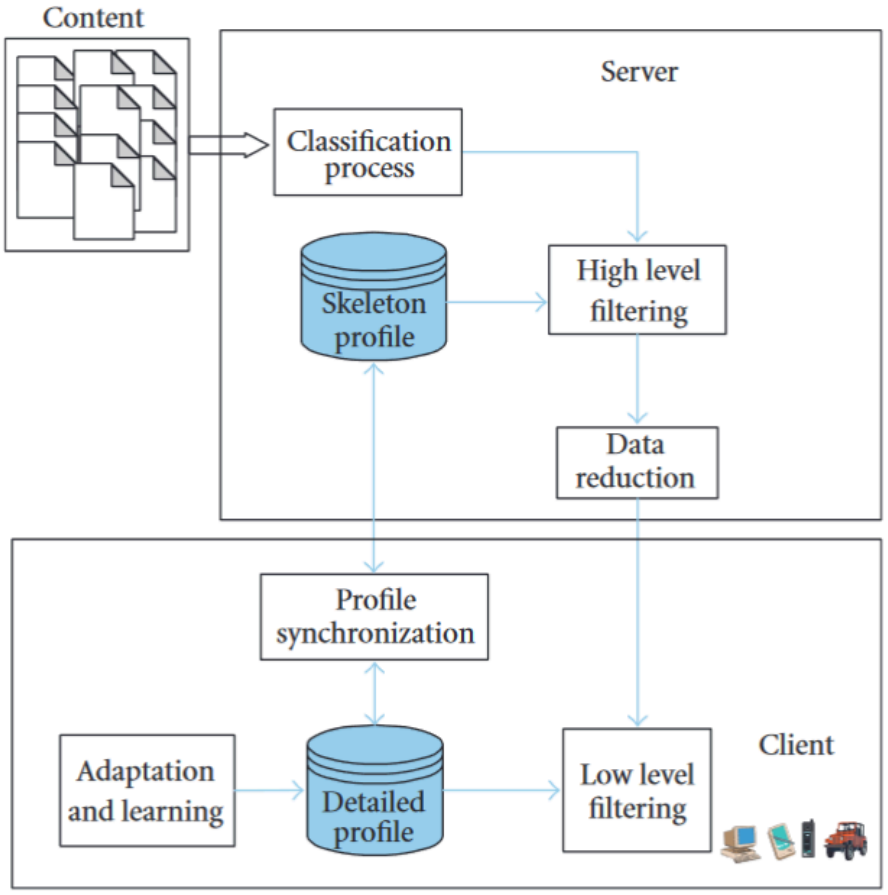


Рисунок 2.2 – Архітектура розподіленої системи

В архітектурному дизайні розподіленої системи визначено чотири ключові елементи, які необхідно брати до уваги. Ці чотири елементи описано нижче:

1. Що таке комунікаційні сутності? Зазвичай це спостерігається для різних рівнів абстракції залежно від фактичного середовища та можливостей обчислювальних пристроїв, що використовуються. У більшості випадків видно, що процеси всередині пристроїв здійснюють зв’язок. У пристроях з дуже обмеженими ресурсами, де абстракція процесів може не існувати в операційній системі, це може бути фактичний пристрій, який називається вузлом, який спілкується. З більш потужними пристроями та більш складними операційними системами процеси можуть бути доповнені за допомогою потоків. Таким чином, потоки будуть комунікаційною кінцевою точкою. З абстракції, орієнтованої на програмування, визначають об’єкти, компоненти або веб-сервіси як потенційні кінцеві точки комунікацій.

2. Яку комунікаційну парадигму використовуватимуть суб’єкти? Парадигми комунікацій, визначені комунікацією [19], включають міжпроцесну комунікацію, віддалений виклик і непряму комунікацію. Міжпроцесовий зв’язок передбачає передачу повідомлень відносно низького рівня, яка виконується між процесами безпосередньо через примітиви передачі повідомлень і виклики API. Віддалений виклик має форму «протоколів запиту-відповіді» (з використанням базових примітивів обміну повідомленнями), віддалених викликів процедур і віддаленого виклику повідомлень. Як міжпроцесний зв’язок, так і віддалений виклик передбачають прямий зв’язок між відправниками та одержувачами повідомлень, дві сторони зазвичай знають про існування інших, і вони обидві повинні існувати одночасно. З іншого боку, при непрямому спілкуванні зв’язку може сприяти третя сторона, і, отже, автор і кінцевий адресат повідомлення не обов’язково повинні існувати одночасно або знати про існування один одного. Техніки для забезпечення цього непрямого зв’язку включають: груповий зв’язок, системи PublishSubscribe, черги повідомлень, кортежні простори і розподілену спільну пам’ять [20 - 21] .

3. Які ролі та обов’язки мають ці суб’єкти в загальній архітектурі? Компоненти (процеси) розподіленої системи взаємодіють один з одним для

досягнення певної корисної мети, наприклад, сеансу чату [22]. Ця мета конкретної розподіленої системи досягається за допомогою зв’язку цього процесу. Під час виконання цього зв’язку об’єкти, що спілкуються, виконують певні ролі, які є фундаментальними для досягнення загальної функції розподіленої системи. Існує дві основні парадигми для розподілу ролей, а саме: парадигма «клієнт-сервер» і парадигма «одноранговий». У парадигмі «клієнт-сервер» одна сутність позначається як клієнт, а інша — як сервер і кожна з них має свої конкретні ролі, коли клієнт запитує певний ресурс або послугу, а сервер отримує й обробляє запит, відповідаючи певним чином. Однак у парадигмі однорангового зв’язку об’єкти, що спілкуються, поділяють однакові ролі, проте вони міняються ролями залежно від потреб конкретного завдання, яке має виконати розподілена система.

4. Як ці взаємодіючі об’єкти фізично розміщені в загальній архітектурі? Це передбачає відображення об’єктів або служб, що спілкуються, із фактичним фізичним апаратним забезпеченням, яке складає розподілену систему. Фізичне апаратне забезпечення зазвичай неоднорідне за своєю природою, тому потрібне ретельне розміщення, оскільки воно впливає на продуктивність, надійність і безпеку всієї розподіленої системи [23]. Розміщення має враховувати загальну кінцеву функціональність системи, а отже, також кілька факторів, таких як шаблони зв’язку між об’єктами, надійність машин з огляду на їх поточне навантаження та якість інтерфейсу зв’язку між пристроями.

Загальні стратегії розміщення включають:

- відображення служб на кількох серверах у вигляді розділів або реплік;
- кешування, тобто зберігання нещодавно використаних даних у певних місцях;
- мобільний код, де код завантажується або надсилається з сервера та виконується на клієнті, як, наприклад, в аплетах;

- мобільні агенти, які являють собою програмні додатки та дані, які переміщуються мережею від одного пристрою до іншого, виконуючи задане завдання та зрештою повертаючи результати призначеній делегуючій організації.

Для цілей цього дослідження, у більшій області розподілених систем, ми детальніше розглянемо парадигми, які керують розробкою та розгортанням однорангових систем і систем розподіленого зберігання.

2.2.1. Однорангові системи

Однорангові системи являють собою парадигму для побудови розподілених систем і додатків, у яких дані та обчислювальні ресурси надаються багатьма хостами в Інтернеті, усі з яких беруть участь у наданні єдиної послуги». На відміну від моделі клієнт-сервер, будь-який вузол у мережі P2P може діяти як сервер для інших і, в той же час, діяти як клієнт. Комунікація та обмін інформацією здійснюються безпосередньо між учасниками, а зв’язки між вузлами в мережі є рівноправними».

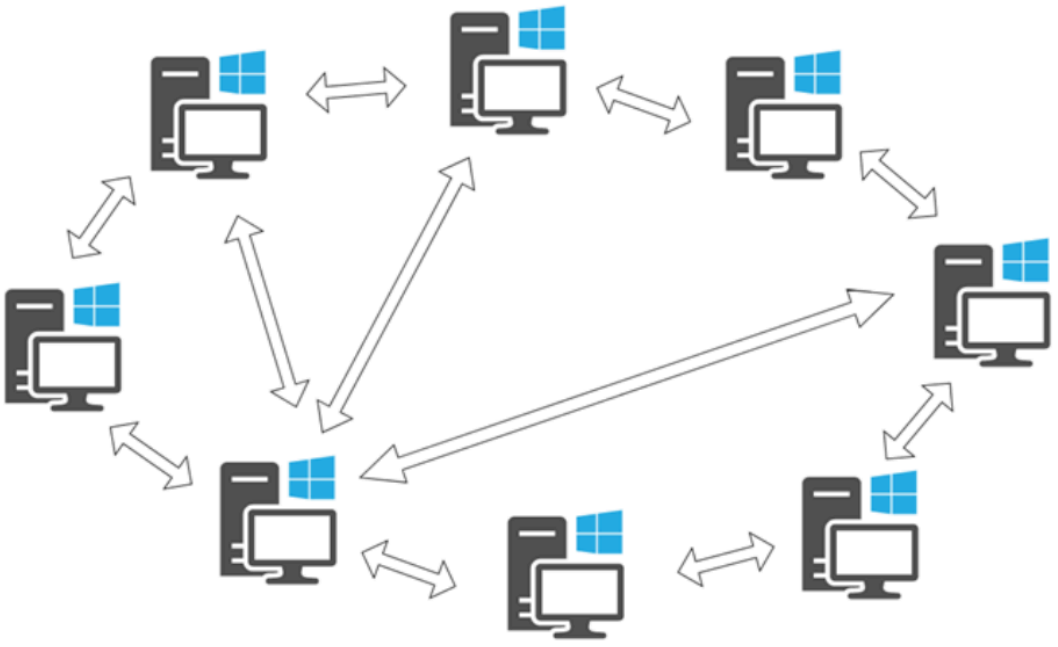


Рисунок 2.3 – Загальна структура однорангової мережі (P2P)

Суть однорангових (P2P) систем полягає в тому, що вони не обов’язково покладаються на централізований сервер для надання конкретної послуги. Вузли, що беруть участь у системі, ділять ролі, і, отже, будь-який вузол у системі P2P може взяти на себе роль сервера залежно від обставин, стану системи чи конкретної дії, яку потрібно виконати. Таким чином, у випадку різкого збільшення числа об’єктів у системі, їм не потрібно покладатися на один перевантажений ресурс для надання необхідної послуги. Усі вузли/пристрої, що беруть участь, також можуть діяти як сервери для обслуговування потреб інших вузлів. Таким чином, системи P2P масштабуються краще, ніж парадигми клієнт-сервер. Крім того, оскільки кілька пристроїв-учасників можуть надавати послугу, а не один сервер, надійність загальної системи може підвищитися порівняно з надійністю послуги, що надається одним сервером. Таким чином, системи P2P допомагають усунути одиничні точки відмови, підвищуючи доступність послуг і, отже, надійність.

Також зазначається, що, оскільки P2P-системи складаються з потенційно великої кількості пристроїв, вони мають можливість створювати порівняно великий об’єднаний пул ресурсів. Це пов’язано з тим, що вони мають можливість спільно використовувати ресурси (сховище, обчислення або зв’язок) із великою кількістю пристроїв на межі мережі.

Хоча системи P2P можуть забезпечити масштабованість, доступність і потенційно великий пул ресурсів, вони також мають певні проблеми, які вони повинні подолати.

Пристрої кінцевої точки, які надають свої ресурси, зазвичай контролюються кінцевими користувачами (а не централізованим органом) і, таким чином, залежать від уподобань користувача. Це означає, що вони можуть значно відрізнятися з точки зору доступності фізичних ресурсів (потужність процесора, оперативна пам’ять, пам’ять, пропускна здатність мережі та заряд акумулятора), платформи операційної системи, програмних

програм і навіть використання. Ця неоднорідність вузлів створює проблему для розміщення ресурсів для досягнення загальної мети системи.

Ще одна проблема, з якою потрібно боротися системам P2P, це можливість випадкового приєднання вузлів до системи та виходу з неї. Цю концепцію зазвичай називають відтоком. Відтік пов'язаний із завантаженням системи, а також із підключенням до мережі. Обчислювальне навантаження пристрою може призвести до втрати відповіді, що може призвести до відключення вузла від мережі.

Використання різноманітних параметрів підключення до мережі, доступних кінцевим пристроям користувача, також може вплинути на підключення. Крім того, використання приватних IP-адрес, трансляція мережевих адрес за брандмауерам і правила брандмауера також можуть впливати на підключення.

Таким чином, системи P2P повинні мати можливість прозоро справлятися з відтоком, щоб користувачі системи не помічали входження нових вузлів або неоголошене зникнення вузлів. Це потенційно може вплинути на доступність системи на обох кінцях шкали: якщо в мережі P2P забагато вузлів або якщо певні вузли періодично зникають з мережі, якість наданої послуги не повинна суттєво знизитися.

Системи P2P зазвичай використовують базову мережеву інфраструктуру та протоколи, які забезпечують маршрутизацію (IP), іменування (DNS), реплікацію даних (UDP), надійність (TCP) і безпеку (HTTP/S). Однак цих уже існуючих механізмів може бути недостатньо, тому вони використовують власну накладену мережеву інфраструктуру з використанням протоколів прикладного рівня, щоб доповнити існуючий і забезпечити більш специфічну функціональність, спрямовану на необхідна спеціальна послуга. Як правило, оверлейна мережа відповідає за керування розміщенням і пошуком ресурсів в межах однорангових вузлів перед лицем загрозливого відтоку, забезпечуючи досягнення загальних цілей системи P2P.

Зазвичай існує 3 типи накладень: структуровані накладання, неструктуровані накладки та гібридні накладки [24].

Структуровані накладення: цей тип накладень використовує організовані та детерміновані форми розміщення ресурсів (включно з упорядкуванням вузлів) і алгоритми пошуку. Ця організація визначається структурними потребами системи для конкретної області застосування. Завдяки такій організованій структурі такі накладки можуть досягти відносно швидшого часу пошуку, однак вони можуть мати недоліки доступності, коли стикаються з високим рівнем відтоку.

Неструктуровані накладення: вони складаються з накладень, які використовують більш вільно сформовані алгоритми, які є детермінованими, але все ще відкритими для забезпечення недетермінованого розміщення ресурсів, керування вузлами та пошуку. Загалом, немає відомого конкретного однорангового вузла або підмножини однорангових вузлів, які мають повні знання про мережу та її ресурси.

Механізми, які тут використовуються, втілюють певну форму випадкової взаємодії або затоплення, щоб полегшити пошук однорангових користувачів або ресурсів. Тривалість пошуку може бути необмеженою, однак надійність загальної системи є кращою, коли вона стикається з високим рівнем відтоку. Також кажуть, що вони підтримують більш складні форми запитів, такі як запити з точним ключовим словом, запити діапазону та запити на основі атрибутів.

Гібридні накладання: ці накладання характеризуються поєднанням методів структурованого накладання, методів неструктурованого накладання або централізованих механізмів клієнт-сервер. Вони мають на меті об’єднати переваги різних форм накладень, а також тих, які можна побачити в парадигмах клієнт-сервер. Масштабованість, надійність і швидкість пошуку зазвичай є основними факторами, які визначають, які служби призначені для структурованої, неструктурованої або централізованої архітектури клієнт-сервера.

2.2.2. Стратегії розміщення ресурсів і механізми взаємодії

У накладеннях P2P, як і в будь-якій іншій розподіленій системі, розміщення ресурсів і стратегії однорангової взаємодії (комунікації) відіграють важливу роль у досягненні цілей самої системи. Структуровані та неструктуровані накладення P2P самі по собі використовують різноманітні стратегії розміщення ресурсів і рамки взаємодії, як обговорюється нижче.

Серед стратегій розміщення інформації про індексування про те, де знайти вузли чи ресурси, загалом існує 3 основні підходи: централізоване індексування, розподілене індексування та локальне індексування. Індексація служить системою адресації, за допомогою якої шукаюча сторона може знайти потрібний ресурс або вузол. Вони можуть впливати на швидкість пошуку, а також на надійність пошуку.

Централізовані стратегії зазвичай передбачають розміщення інформації індексування на одному центральному сервері або невеликій групі серверів.

Розподілені стратегії передбачають розподіл індексів між одноранговими партнерами, що беруть участь у системі P2P. Це може включати розділення та реплікацію. Одним із поширених методів увімкнення розподіленого індексування є розподілені хеш-таблиці (DHT), які зазвичай можна побачити в структурованих накладаннях. Тут пари ключ-значення складають записи хеш-таблиці, а частини цієї таблиці розподіляються між рівноправними користувачами. Деякий ідентифікатор вузла або ресурсу хешується для формування ключа, і певна форма адресної інформації призначається ключу. Алгоритми DHT відрізняються за своїми схемами реалізації з точки зору техніки геш-відображення, використовуваних алгоритмів пошуку, діаметра пошуку, діаметра пошуку, відмовостійкості та стійкості до відтоку.

Стратегії локального індексування передбачають реєстрацію та розміщення окремими вузлами лише локальних ресурсів або об’єктів даних. Реплікація може виконуватися локально залежно від параметрів алгоритму.

Знання про віддалені ресурси або об’єкти даних зазвичай потрібно шукати в мережі P2P. Ця стратегія зазвичай використовується в неструктурованих накладеннях у поєднанні з алгоритмами затоплення та «випадкового блукання» для полегшення пошуку. Структура таких накладень не визначена заздалегідь, однак деякі природні структури, такі як випадкові графи степеневого закону, де ступені вузлів відповідають степеневому закону; або нормальні випадкові графи, де вузли зазвичай мають однакові ступені.

Структура взаємодії визначає вибір, який робить одноранговий партнер, з яким йому слід спілкуватися, щоб отримати певний результат пошуку або отримати послугу.

Існує 2 основні категорії структур взаємодії: рамки випадкової взаємодії і рамки детермінованої взаємодії:

1. Фреймворки випадкової взаємодії передбачають випадковий вибір однорангового або набору однолітків для спілкування. Зазвичай це відбувається у формі затоплення, використання «k-випадкових ходунів». Флуд передбачає сліпу трансляцію повідомлень, у той час як використання «k-random walkers» передбачає випадковий вибір початкової групи для початку передачі повідомлень для виконання пошуку, у свою чергу, кожен випадковий відвідувач також вибирає випадкового однорангового або набір однорангових пристроїв для спілкування. Щоб уникнути проблеми необмеженості повідомлень і надмірного переповнення системи, ці стратегії можуть використовувати механізм перевірки, коли під час кожного переходу одноранговий вузол перевіряє автора повідомлення. Повідомлення з часом життя (TTL) також можуть використовуватися, щоб переконатися, що повідомлення не потрапляють у нескінченні цикли. Останньою стратегією, яку вони можуть використати, є використання стратегії «кільця, що розширюється», коли спочатку заливають або консультують лише невелику групу, а потім, якщо це не вдається, використовується поступово більша група, доки не буде досягнутий результат пошуку. Рамки випадкової взаємодії частіше використовуються в неструктурованих накладаннях.

2. Структури детермінованої взаємодії передбачають використання алгоритмічного рішення для передбачуваного рішення, з яким партнером взаємодіяти. На це рішення може вплинути хеш певного ідентифікатора ресурсу (у випадку DHT), розташування вузла або деякі властивості вузла (включно з пропускною спроможністю, ємністю диска, зарядом акумулятора, часом безвідмовної роботи або поточним навантаженням тощо).

Існує 3 стратегії серед структур розподіленої взаємодії, а саме: рівномірний розподіл, ієрархічна взаємодія та механізми публікації-підписки. За допомогою рівномірного розподілу метою є досягнення балансування навантаження між одноранговими вузлами, таким чином адресний простір і ресурси розподіляються рівномірно між одноранговими вузлами-учасниками. Отже, діаметр пошуку має тенденцію бути узгодженим у мережі P2P для всіх взаємодій. При ієрархічній взаємодії певний набір вузлів у мережі призначений для виконання особливої ролі супер-однорангових вузлів. Таким чином взаємодія більше спрямована на супероднолітків. Така стратегія вимагала б розумного вибору супероднорангів на основі набору критеріїв, а також заміни супероднорангів, якщо вони покинуть мережу. Нарешті, форми взаємодії «публікація-підписка» характеризуються розподілом обов’язків на основі потреби в інформації. Певні однолітки відіграють роль постачальника інформації (джерела) або розповсюдження, тоді як інші виконують роль споживача (споживача). Споживачі прагнуть підписатися на певні джерела, а джерела надсилають інформацію лише тим споживачам, які підписалися на отримання інформації. Таким чином, ця структура взаємодії зазвичай асоціюється з системами сповіщення про події або системами розповсюдження повідомлень.

2.2.3. Великомасштабні розподілені системи зберігання

В епоху інформації створення великих обсягів цифрових даних призвело до потреби у збільшенні місткості пристроїв зберігання. Для того,

щоб не відставати від необхідності зберігати великі обсяги даних, було застосовано дві основні стратегії.

Перша – це стратегія вертикального розширення, коли фізична ємність пристроїв зберігання з часом збільшується відповідно до закону Мура. Запам'ятовуючі пристрої, такі як оптичні носії та флеш-пам'ять, як правило, слідують цій тенденції збільшення розміру, однак вони все ще не в змозі встигати за поточним надлишком даних.

Друга застосована стратегія – стратегія горизонтального розширення. Саме тут ємність зберігання з кількох пристроїв була об’єднана, щоб діяти як єдиний об’єднаний ресурс за допомогою методів розподілених систем. Двома основними результатами цієї стратегії горизонтального розширення є розподілені файлові системи та розподілені бази даних. У той час як файлові системи добре працюють з неструктурованими даними, бази даних працюють краще зі структурованими даними. Це фундаментальна відмінність між файловими системами та базами даних.

Розподілені файлові системи. Спочатку файлові системи створювалися для окремих обчислювальних систем як можливість операційної системи полегшувати доступ до програмного інтерфейсу для зберігання даних на диску. Спочатку їм потрібно було лише підтримувати постійність даних, забезпечувати можливості доступу, надійність і продуктивність для кількох програм на одному пристрої. У зв’язку з потребою в обміні даними через мережі виникли подальші відповідні вимоги, включаючи контроль доступу, доступність, узгодженість і балансування навантаження серед інших. Нижче наведено деякі важливі вимоги до розподілених файлових систем:

- Прозорість передбачає приховування основної діяльності в ядрі розподіленої системи, щоб користувач не помічав змін, що відбуваються під капотом. Прозорість розглядається з кількох аспектів, включаючи: прозорість доступу, коли користувач або його програми мають уніфікований метод доступу до функцій файлової системи; Прозорість розташування/міграції , де забезпечується єдиний механізм адресації незалежно від переміщення файлів

між вузлами; Прозорість продуктивності , яка вимагає, щоб на якість обслуговування користувача не впливали зміни кількості вузлів або їх відповідного завантаження.

- Паралельність вимагає, щоб файл, який змінює один користувач, не впливав на той самий файл, який змінює інший користувач.
- Реплікація передбачає наявність копій одного файлу в кількох місцях. Це покращує продуктивність, оскільки кілька вузлів можуть надавати послуги цьому файлу, а також може забезпечити відмовостійкість, якщо певний вузол із копією файлу стане недоступним.
- Неоднорідність апаратного забезпечення та ОС необхідно вирішити таким чином, щоб програмне забезпечення, що підтримує розподілену файлову систему, можна було розгорнути на різноманітних платформах апаратного забезпечення та операційних систем. Це допомагає збільшити охоплення файлової системи в різних середовищах.

Відмовостійкість – це здатність служби протистояти збоям, таким чином підтримуючи нормальну або близьку до нормальної роботи.

- Узгодженість особливо важлива в системах, які дозволяють множинний доступ до одного файлу або копій файлу. Це вимагає, щоб незалежні модифікації призводили до єдиної уніфікованої версії без відхилень, що є результатом паралельних модифікацій.
- Безпека важлива, оскільки розповсюдження файлів забезпечує більшу доступність. Таким чином, цей доступ має бути обмежений лише авторизованими сторонами. Таким чином, у деяких випадках можуть знадобитися механізми автентифікації, політики контролю доступу, механізми перевірки цілісності та шифрування.
- Ефективність є важливим фактором, оскільки розподілені файлові системи повинні працювати принаймні на тому ж рівні, що й звичайні файлові системи, щоб бути корисними.
- Масштабованість - це здатність системи змінювати розмір залежно від навантаження або кількості вузлів. Враховуючи більшу кількість

вузлів або збільшення потреби в просторі для зберігання, система повинна сприяти легкому розширенню та бути здатною задовольняти зростаючі потреби без зниження рівня обслуговування.

Було розроблено кілька різних розподілених файлових систем з різними характеристиками залежно від конкретних потреб користувача чи організації. Деякі з відомих включають Google FileSystem (GFS), GlusterFS, Ceph, Hadoop Distributed Filesystem (HDFS) і XtreemFS [24 - 27]. Декілька інших, безсумнівно, будуть розроблені для задоволення потреб інших більш конкретних користувачів. Архітектури розповсюдження файлів P2P у деяких випадках класифікуються як форма розподілених файлових систем, однак вони не є файловими системами як такі, оскільки багато з них не пропонують звичайний API файлової системи, і, крім того, їх кінцевою метою є висока доступність файлів у великих мережах.

2.2.4. Розподілені бази даних

Розподілена база даних — це сукупність кількох логічно взаємопов’язаних баз даних, розподілених через комп’ютерну мережу [28]. Важливими факторами тут є те, що розділи більшої бази даних логічно взаємопов’язані і що єдиний засіб зв’язку між ними розділів через комп’ютерну мережу. Логічний взаємозв’язок означає, що існує реляційний зв’язок між елементами даних, пов’язаними через фізичне розділення. Крім того, єдиним фізичним з’єднанням, що об’єднує розділені розділи, є комп’ютерна мережа. Таким чином, бази даних, які спільно використовують лише пам’ять, дискове сховище або процесори, не вважаються розподіленими базами даних, навіть якщо вони можуть бути розділеними.

Розподілені бази даних зазвичай відповідають тим же вимогам, що й розподілені файлові системи, описані раніше. Однак вони більше спрямовують ці вимоги на необхідність обробки транзакцій бази даних і

запитів. Виділяють такі функціональні можливості, необхідні для розподілених баз даних:

- Управління розподіленим каталогом/каталогом: це передбачає відстеження розподілу даних, фрагментації, реплікації та пов’язаних метаданих розподіленої бази даних.
- Розподілена обробка запитів: це дозволяє отримати доступ до даних шляхом передачі запитів у кілька віддалених місць і агрегувати очікувану відповідь.
- Управління розподіленими транзакціями: це передбачає передачу запитів на маніпулювання даними в кількох віддалених розділах розподіленої бази даних, щоб синхронізувати ці зміни між даними в усій системі та забезпечити цілісність транзакції та зачеплених даних у цілому.
- Управління реплікованими даними: це передбачає вибір того, до якої копії реплікованих даних отримати доступ, а також збереження узгодженості інших копій.
- Відновлення розподіленої бази даних. Критично важливим для баз даних є відновлення після різних типів збоїв, включаючи збої транзакцій та збої мережі.

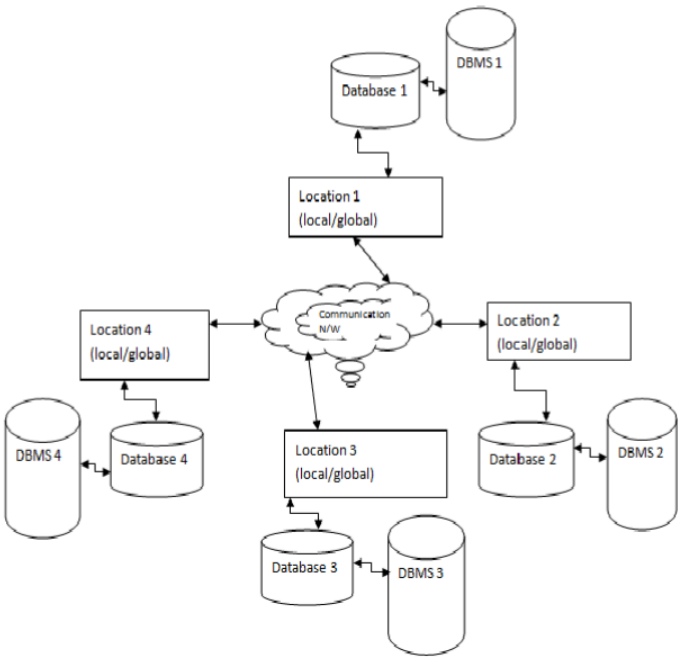


Рисунок 2.4 – Архітектура розподіленої СУБД

Додатково класифікують розподілені бази даних за трьома критеріями: ступінь розподілу, ступінь автономності та ступінь однорідності. Ступінь розподілу визначає, чи буде база даних повністю чи частково розподілена. Ступінь автономії визначає, чи може окремий локальний сайт працювати як незалежний суб'єкт за певних обставин. Ступінь однорідності визначається тим, чи всі локальні сайти використовують однакове програмне забезпечення СУБД, чи деякі з них відрізняються з різних причин.

З цього критерію вони описують 3 типи розподілених баз даних:

- Системи чистих розподілених баз даних, у яких база даних представлена через одну кінцеву точку та існує єдина схема бази даних. База даних здається користувачеві централізованою базою даних, оскільки вона не має автономності через єдину кінцеву точку.
- Системи об'єднаних баз даних мають єдину схему бази даних, однак кожен окремий вузол/локальний сайт може функціонувати автономно, взаємодіючи з клієнтами та незалежно обробляючи транзакції.
- Багатобази даних/однорангові системи баз даних функціонують автономно, взаємодіючи з клієнтами та незалежно обробляючи транзакції. Крім того, вони не мають попередньо визначеної глобальної уніфікованої схеми бази даних, спільної з іншими однолітками. Вони створюють схему на вимогу залежно від програми, яка запитує дані або виконує транзакцію.

Як розподілені файлові системи, так і розподілені бази даних виникли внаслідок потреби зберігати більші обсяги даних. Однак через різні форми даних (структуровані та неструктуровані), а також їхні специфічні індивідуальні вимоги спонукали до окремого розвитку технологій. З часом критерії, що розділяють ці два, розмилися, і в деяких випадках вони злилися, що призвело до форми сховищ даних NoSQL.

Сховища даних NoSQL мають на меті покращити основну проблему масштабованості, яку мають реляційні бази даних, коли вони стикаються з величезними обсягами даних. акі типи сховищ даних не відповідають суворо властивостям ACID реляційних баз даних, скоріше вони відповідають більш

вільному набору обмежень, які називаються BASE (Bally, Available , Soft State, Eventual Consistency). По суті, використовуючи теорему CAP (консистенція, доступність, стійкість розділу), реляційні бази даних, що дотримуються властивостей ACID, більше зосереджуються на досягненні узгодженості та доступності, намагаючись підтримувати якомога менше розділення даних. Сховища даних NoSQL, намагаючись досягти розподілу, більше зосереджуються на досягненні стійкості розділів і доступності з меншими вимогами до узгодженості. Таким чином бази даних NoSQL досягають кращого масштабування через розповсюдження, однак вони або не мають глобально уніфікованої схеми, або генерують схему на вимогу.

Існує 5 основних типів сховищ даних NoSQL, які визначаються основними характеристиками самих сховищ даних:

- Ключ-значення (кортеж) сховища даних, за допомогою яких дані індексуються та адресуються за допомогою хешу даних таким чином, що хеш і фактичні дані утворюють пару (пару ключ-значення). Такі сховища даних часто не відповідають певній схемі бази даних. Вони пропонують високу масштабованість і швидкий пошук.
- Прикладами таких сховищ даних є Dynamo, Riak і Redis.
- Сховища даних, орієнтовані на стовпці, приблизно імітують концепцію рядків і стовпців, як це видно в традиційних реляційних базах даних. Кожен ключ пов’язаний із набором атрибутів, які нагадують окремі стовпці. Різні структури даних використовуються для підтримки стовпчастих даних у різних місцях. Прикладами таких сховищ даних є Bigtable, HBase і Cassandra.
- Сховища документів використовують концепцію документа, який містить певну форму кодування всередині нього, де кожен документ вільно представляє запис бази даних, що містить атрибути. Кожен документ може мати власну схему кодування. Поширені кодування включають XML, YAML, JSON і двійковий JSON. Прикладами такого типу сховища даних є MongoDB і CouchDB.

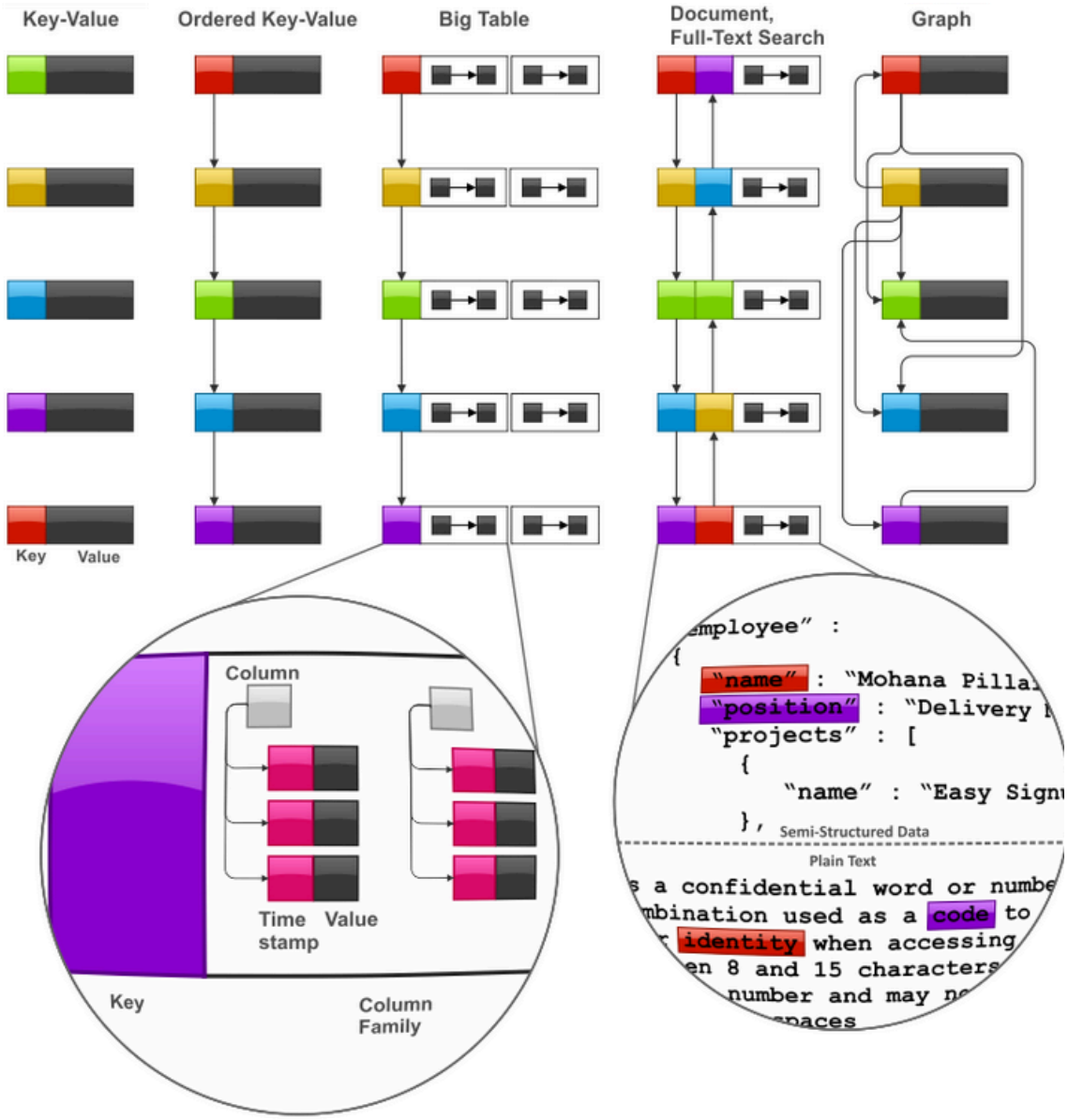


Рисунок 2.5 – Основні типи NoSQL баз даних

- Графові бази даних зберігають дані у формі вузлів і ребер. Вузли представляють об’єкти даних, а зв’язки – зв’язки між вузлами даних. Ребра можуть бути спрямованими або ненаправленими. Вузли можуть містити атрибути як листові вузли або можуть містити атрибути як властивості самого вузла залежно від реалізації бази даних. Вони добре працюють із напівструктурованими даними, які за своєю суттю мають зв’язки між такими елементами даних, як соціальні мережі, мережі дорожнього руху чи географічні карти. Вони пропонують масштабованість, швидкий пошук,

сумісні з ACID і пропонують підтримку відкату. Приклади графових баз даних включають Neo4j, Orient DB і Ontotext Graph DB.

2.3 Мобільні та хмарні середовища як вид розподілених систем

Визначено 4 тенденції, які зараз впливають на розподілені системи та обчислювальні технології в цілому. Ці чотири тенденції включають:

- Поява всеохоплюючої мережевої технології через наявність пристроїв із високим ступенем взаємозв'язку, які дозволяють підключатися в будь-якому місці та в будь-який час через Інтернет.
- Поява повсюдних обчислень, що підтримується мобільністю в розподілених системах, а також мініатюризацією та портативністю обчислювальних пристроїв. Це призвело до того, що обчислювальні пристрої присутні майже всюди.
- Зростаючий попит на розподілені мультимедійні системи, тобто здатність розподіленої системи підтримувати зберігання, передачу або представлення різноманітних типів медіа (наприклад, тексту, зображень, відео та аудіо) інтегрованим способом. Широке застосування цього можна побачити в доступі до прямих або записаних трансляцій, доступі до фільмотек, які пропонують послуги відео на замовлення, доступі до музичних бібліотек і засобів відеоконференцій.
- Погляд на розподілене обчислення як на корисність або товар, де ресурси, надані відповідним постачальником, можна купувати чи орендувати частинами за потреби. Це породило концепцію хмарних обчислень.

Як мобільне, так і хмарне середовище включають комп'ютерні пристрої, які допомагають користувачам виконувати різноманітні дії, які можуть включати спілкування, банківські послуги, бізнес і доступ до медичних послуг, а також державних послуг. У наступних підрозділах ми розглядаємо основні аспекти мобільних пристроїв і середовищ хмарних обчислень.

Мобільні та вбудовані пристрої часто схожі на клієнтську сторону спектру, де вони пропонують відносно меншу кількість обчислювальної потужності в портативній формі на краях мережі, безпосередньо для користувача. Датчики та інші менші вбудовані пристрої часто мають ще менші обчислювальні можливості, щоб зробити їх дуже портативними, непомітними та підключаючими до інших пристроїв.

Через відносно обмежені обчислювальні можливості мобільні та вбудовані пристрої часто є кінцевими пристроями, які споживають послуги, що надаються іншими. Вони також можуть служити точками збору або зворотного зв'язку, які отримують інформацію з країв мережі для того, щоб її відправити назад у серверну частину для подальшої обробки.

Смартфони пристосувалися до різних цілей, як згадувалося раніше, однак, окрім локальних обчислень на фактичному пристрої, деякі з їхніх основних цілей пов'язані з їхніми пристроями введення/виведення, точніше підключенням до мережі та наявними датчиками. Можливо, найпоширенішими формами підключення до мережі, які активно використовуються в смартфонах, є стільниковий радіозв'язок, WiFi і Bluetooth.

У своїй нормальній роботі смартфони підключаються через свою стільникову радіостанцію до постачальника послуг мобільної мережі за допомогою набору базових станцій, реєстрів користувачів і внутрішніх служб з'єднання, щоб полегшити мобільний зв'язок. Вони також можуть використовувати ту саму інфраструктуру стільникової мережі для доступу до служб мобільного передавання даних, за допомогою яких вони безперешкодно підключаються до Інтернету через протоколи стільникової мережі. Смартфон також може підключитися до Інтернету через свій радіоінтерфейс WiFi. З'єднання спрямовується до власної мережевої інфраструктури WiFi користувача або до третьої сторони, яка надає точку доступу до структурованої або спеціальної мережі, де надається підключення до Інтернету або якась інша мережева послуга. Нарешті, смартфони можуть

з’єднуватися через радіо Bluetooth один з одним або з іншими периферійними пристроями, такими як клавіатури, пристрої «вільні руки», динаміки або будь-які інші пристрої з підтримкою Bluetooth, щоб покращити їх функціональність. З’єднання NFC також може бути ще одним варіантом, коли мобільні пристрої зв’язуються з іншими пристроями або трансиверами, щоб отримувати послуги або покращувати функціональність на дуже близькій відстані.

За допомогою цих параметрів підключення до мережі смартфон може взаємодіяти з:

- постачальником послуг мобільного зв’язку, який надає послуги стільникового зв’язку та передачі даних в Інтернеті (також може бути присутнім проміжний постачальник інфраструктури);
- постачальником операційної системи, що надає оновлення ОС і нові функції;
- постачальником обладнання, що надає оновлення драйверів;
- іншими постачальниками послуг додатків;
- іншими користувачами у відповідному мережевому середовищі.

Ці сторони можуть надавати законні послуги, як описано вище, або вони можуть мати зловмисні мотиви, які суперечать цілям користувачів.

2.4 Моделі послуг та сутності, що надаються хмарними розподіленими інфраструктурами

Хмарні обчислювальні середовища в основному походять із серверного кінця спектра, де часто є величезна кількість доступної обчислювальної потужності, однак вони також поширюються на клієнтську сторону спектру, де хмарні служби (такі як служби зберігання даних, обчислювальні служби) використовуються як корисні. продукти окремих користувачів або організацій.

Національний інститут стандартів і технологій (NIST) визначає хмарні обчислення як «модель забезпечення повсюдного, зручного мережевого доступу на вимогу до спільного пулу конфігурованих обчислювальних ресурсів (наприклад, мереж, серверів, сховищ, програм і служб) які можна швидко підготувати та випустити з мінімальними зусиллями адміністратора або взаємодії постачальника послуг». Вони також визначають п’ять основних особливостей, які характеризують хмарні обчислення:

- Самообслуговування на вимогу: це означає, що споживач може в односторонньому порядку придбати обчислювальні ресурси (обчислювальну техніку, сховище тощо), як того вимагають його потреби, без обов’язкової взаємодії людини з постачальником послуг.
- Широкий мережевий доступ: це стосується можливості доступу до ресурсів через мережі, а також сприяння доступності через стандартні вже прийняті та гетерогенні платформи чи пристрої.
- Об’єднання ресурсів: це передбачає об’єднання ресурсів постачальником послуг для обслуговування кількох споживачів у моделі з кількома клієнтами, тобто, коли споживачі спільно використовують пул ресурсів. Цей пул віртуальних і фізичних ресурсів із різних місць динамічно розподіляється та перерозподіляється між споживачами залежно від варіацій їх попиту та потреб у продуктивності.
- Швидка еластичність: це означає, що ресурси, які постачальник хмарних послуг надає споживачам, можуть прозоро масштабуватися вгору або вниз залежно від потреб користувачів, не впливаючи на інших користувачів. Таким чином, хмарні послуги часто здаються нескінченно багатими для більшості окремих споживачів.
- Вимірювана послуга: ця можливість передбачає прозорий і детальний моніторинг і вимірювання розподілу та використання наданих ресурсів. Таким чином, зібрана інформація може бути використана як постачальником, так і споживачем для оптимізації розподілу ресурсів, а

також для контролю використання та вимірювання кількості, що використовується для визначення витрат або належної винагороди.

На додаток до цього, NIST також визначив чотири загальні моделі послуг, які зображують, як хмарні обчислення упаковані для потенційних споживачів. Ці моделі обслуговування:

- Програмне забезпечення як послуга (SaaS): Ця модель послуг надає користувачеві інтерфейс до програмного додатку, як правило, через веб-браузер. Споживач має доступ лише до функцій, доступних у програмі. Конфігурація програми також може бути обмежена відповідно до умов постачальника послуг. Користувачі не можуть керувати, контролювати, налаштовувати чи змінювати будь-яку базову платформу чи компоненти інфраструктури, включаючи операційні системи, сховище, обчислювальну чи мережеву інфраструктуру. Прикладами поширених пропозицій програмного забезпечення як послуги є Документи Google і Microsoft Office 365.
- Платформа як послуга (PaaS): у цій моделі послуг споживачам надається можливість розгортати власні або сторонні програмні додатки, розроблені з використанням мов програмування, бібліотек, служб та інструментів, наданих постачальником послуг. Постачальник послуг, по суті, надає середовище розробки та розміщення програмного забезпечення, де користувачі можуть змінювати конфігурації програмного забезпечення або саму програму. Користувачам не надається жодного контролю над базовою хмарною інфраструктурою, включаючи фізичне обладнання, сховище, мережу чи вибір чи конфігурацію операційної системи. Серед типових прикладів пропозицій платформи як послуги є Heroku та Google AppEngine.
- Інфраструктура як послуга (IaaS): це, мабуть, найбільш гнучка модель служби, яку можна налаштувати. Можливості, надані користувачеві, передбачають надання обчислювальної техніки, сховища, мережі та, можливо, інших поширених, але важливих ресурсів, таких як безпека та моніторинг використання, залежно від їхніх потреб. Користувач не має фактичного контролю над базовою хмарною інфраструктурою, яка підтримує

послугу, однак він має можливість надавати пристрої різного обсягу обчислювальної потужності, пам'яті, сховища та мережевих можливостей. Вони також можуть вибрати операційні системи та програми, які вони хочуть розгорнути на створених пристроях. Прикладами широко використовуваних пропозицій інфраструктури як послуги є Amazon Web Services (AWS), OpenStack, CloudStack і Microsoft Azure.

Одним із найбільш помітних критеріїв, які можна побачити в різних моделях послуг, є обсяг контролю та відповідальності, які має споживач (користувач), у порівнянні з тими, які має постачальник хмарних послуг. Можна побачити, що з SaaS контроль і відповідальність споживача відносно обмежені, тоді як, з іншого боку, постачальник хмарних послуг контролює більшу частину інфраструктури, а отже, і більшу частину відповідальності. Можна помітити, що з PaaS контроль і відповідальність ковзає трохи більше до споживача, а з хмарним розгортанням IaaS велика відповідальність за те, що насправді робить обчислювальна інфраструктура, тоді як постачальник послуг лише гарантує, що обчислювальні ресурси доступні та готові для використання. Цей зв'язок представлено на рис. 2.6 з еталонної архітектури хмарних обчислень NIST.

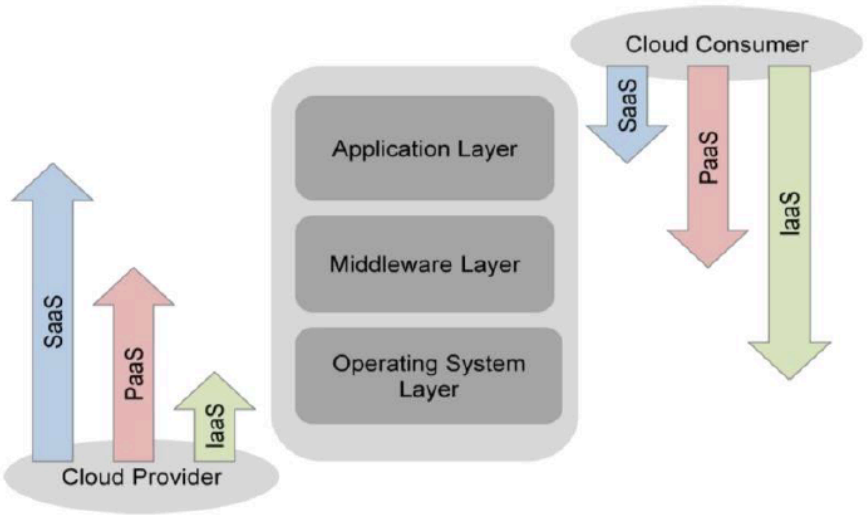


Рисунок 2.6 – Взаємозв'язок контролю та відповідальності між Cloud постачальником послуг і споживачем

NIST також визначає 4 загальні моделі розгортання хмарних інфраструктур. Вони описують, як послуги хмарних обчислень надаються в різних середовищах залежно від потреб користувачів або організації. Моделі розгортання:

- Громадська хмара: у такій хмарній інфраструктурі послуги надаються широкій громадськості для використання для власних обчислювальних потреб. Сервісом може володіти, керувати та керувати організація, наукова установа, державний орган або їх поєднання. Базова хмарна інфраструктура, як правило, знаходиться в приміщеннях постачальника послуг.
- Приватна хмара: цей тип розгортання хмарної інфраструктури надається виключно для окремої організаційної одиниці для внутрішніх або пов’язаних з клієнтом обчислювальних потреб. Сама інфраструктура може бути резидентною на території власника або надаватися виключно для організації третьою стороною.
- Хмара спільноти: це менш поширена модель. Його розгортають для групи зі спільними особливими інтересами та особливими проблемами, які можуть охоплювати організації, наукові установи, уряди чи окремих осіб. Таке розгортання може належати, контролюватися та управлятися однією чи декількома залученими організаціями або третьою стороною. Він також може бути резидентом на території залучених сторін або на території третьої сторони.
- Гібридна Хмара: ця модель розгортання складається з двох або більше попередніх моделей розгортання. Об’єднана інфраструктура працює разом як одиниця завдяки взаємопов’язаним програмним застосункам, які підтримують фактичні бізнес-цілі, однак інфраструктури хмарного розгортання нижче можуть відрізнятися. Типовий приклад цього трапляється, коли організація, яка використовує приватну хмару, хоче пружно масштабуватися за межі локальної потужності, фактично не інвестуючи у фактичну інфраструктуру.

2.5 Представлення моделі хмарної інфраструктури

У середовищі хмарної інфраструктури є кілька суб’єктів, які можуть бути задіяні від надання послуги до надання послуги користувачам. Виділяють 5 основних дійових осіб у просторі хмарного середовища: хмарні споживачі, хмарні провайдери, хмарні аудитори, хмарні брокери та хмарні перевізники.

Споживачі хмари складаються з організацій і окремих осіб, які користуються послугами, що надаються постачальником хмари. Сюди можуть входити окремі особи, працівники компанії, IT-технічний персонал, розробники програмного забезпечення, системні адміністратори та мережеві адміністратори та інші. Залежно від моделі обслуговування, вони можуть взаємодіяти з фактичними бізнес-додатками, платформами розробки програмного забезпечення або інтерфейсами, які дозволяють конфігурувати програмне забезпечення, апаратне забезпечення, мережі та ресурси зберігання. Споживачі, як правило, стурбовані якістю послуг, які вони отримують, а також безпекою та конфіденційністю своїх даних. З цієї причини угоди про рівень обслуговування та угоди про конфіденційність є критично важливими.

Хмарні постачальники несуть відповідальність за придбання необхідних ресурсів і керування програмним забезпеченням, яке забезпечує надання необхідних послуг споживачам. Це може включати технічне обслуговування, конфігурацію та керування апаратним і програмним забезпеченням, яке сприяє хмарним службам. По суті, діяльність провайдера включає розгортання послуг, оркестрування, управління програмним і апаратним забезпеченням, а також вирішення питань безпеки та конфіденційності.

Хмарні аудитори – це зовнішні сторони, які можуть виконувати незалежний аналіз впровадження засобів контролю, відповідності стандартам, а також рівня продуктивності.

Хмарні брокери – це організації, які можуть виступати посередниками між хмарними провайдерами та споживачами хмарних технологій. Через складність хмарних служб хмарні брокери можуть використовуватися для встановлення ділових відносин із клієнтами шляхом збільшення вартості вже існуючих хмарних служб. Це можна зробити шляхом безпосереднього додавання іншої послуги поверх уже існуючої послуги або шляхом об'єднання кількох служб разом від одного чи кількох хмарних постачальників у новий продукт, який може зацікавити потенційних клієнтів.

Хмарні перевізники – це організації, які діють як інфраструктурні посередники, щоб допомогти в доставці та розповсюдженні хмарних послуг споживачам хмарних технологій. Зазвичай вони надають мережеві та телекомунікаційні послуги, через які можна отримати доступ до хмарних служб. Хмарному провайдеру необхідно переконатися, що оператор дотримується угоди про рівень обслуговування, укладеної між двома сторонами, а також забезпечити наявність відповідних механізмів безпеки.

Взаємодія в наданні хмарних послуг між сторонами, описана вище, зображена на рисунку 2.7.

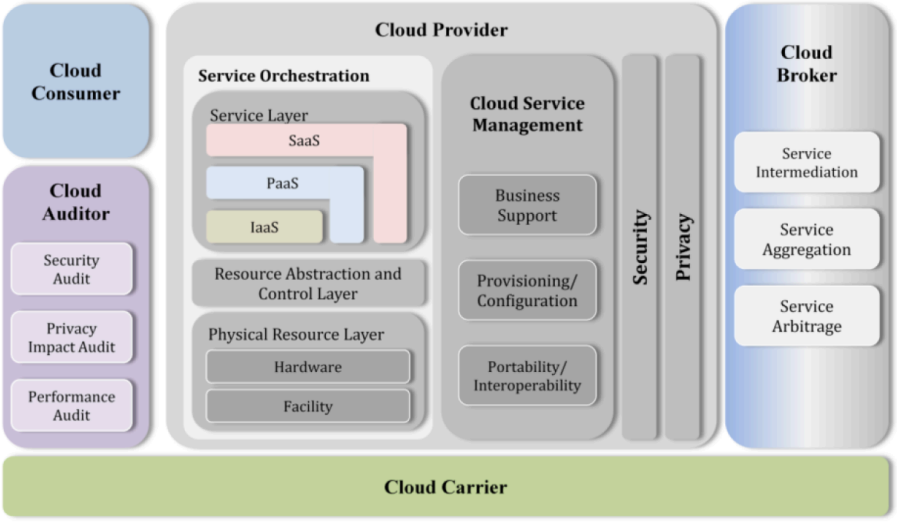


Рисунок 2.7 – Ключові сутності в типовій моделі хмарної інфраструктури

2.6 Процеси віртуалізації та гіпервізори в хмарних розподілених інфраструктурах

Термін «віртуалізація» використовується для широкого опису «відокремлення ресурсу або запиту на послугу від основної фізичної доставки цієї послуги». Технології віртуалізації дозволяють абстрагувати фізичні ресурси (обчислювальні, сховище та мережеві ресурси) у віртуалізовані, щоб забезпечити краще використання та спільне використання фізичних ресурсів, а також гнучкість для легкої адаптації інфраструктури до організаційних змін. Віртуалізація також використовувалася для полегшення консолідації серверів, економії витрат на електроенергію, ізоляції гостьових операційних систем, легкого переміщення системи та полегшення налагодження програмного забезпечення.

Віртуалізація апаратного забезпечення є однією з основних технологій, яка забезпечує хмарні обчислення, оскільки вона дозволяє абстрагувати фізичні апаратні компоненти в більш гнучкі віртуальні екземпляри ресурсів, які можна гнучко розподіляти залежно від різноманітних потреб користувачів. Рівень віртуалізації втілено як «рівень абстракції ресурсів і керування» на рис. 2.7, що зображує сутності в моделі хмарної інфраструктури.

На додаток до віртуалізації окремих підкомпонентів, таких як пам’ять, диски та мережеві інтерфейси, цілі обчислювальні системи також можуть бути віртуалізовані у те, що називається віртуальною машиною. Це те, що називається віртуалізацією платформи або віртуалізацією системи. Щоб реалізувати віртуалізацію системи, між фактичною фізичною машиною та рівнем платформи операційної системи вводиться програмний рівень для підтримки кількох архітектур. Програмний рівень, який забезпечує віртуалізацію системи, зазвичай називають «керуючою програмою», монітором віртуальної машини (VMM) або гіпервізором.

Гіпервізор описується як високопривілейована частина програмного забезпечення, яка працює разом або під керуванням операційної системи та розроблена як «ефективна ізольована копія реальної машини». Відповідно існує 2 типи гіпервізорів:

- Тип I – гіпервізори (гіпервізори Bare-metal): цей тип гіпервізора працює безпосередньо на апаратному забезпеченні. Це основна система завантаження апаратного забезпечення, яка працює на найвищому рівні привілеїв. Він має повний контроль над будь-якою віртуальною машиною, яка працює на ньому. Прикладами такого типу гіпервізора є VMware ESX/ESXi, Microsoft Hyper-V і Citrix XenServer.
- Тип II – гіпервізори (розміщені гіпервізори): цей тип гіпервізора розташовується поруч або працює поверх іншої операційної системи. Він може використовувати драйвери з операційної системи хостингу для доступу до пристроїв введення/виведення. Крім того, він може мати власні вбудовані драйвери, які проксі-система звертаються до апаратного забезпечення через операційну систему хоста. Приклади цього типу гіпервізора включають VMware Workstation, VMWare Player, Virtualbox і QEMU

Хоча ця відмінність зроблена для класифікації технологій віртуалізації, існують певні гіпервізори, такі як KVM (Kernel Virtual Machine) та інші подібні технології, такі як контейнери (наприклад, Docker і OpenVZ), які не обов’язково дотримуються цієї відмінності. Таким чином, вони не підпадають під якусь конкретну категорію, оскільки їх можна впроваджувати безпосередньо на «голому стані» або розміщувати в уже існуючій операційній системі.

На додаток до розподілу та планування фізичних ресурсів між віртуальними машинами, ключовою роллю гіпервізора є забезпечення рівня абстракції, який діє як уніфікований інтерфейс для посередництва між програмним забезпеченням, створеним для різних комп’ютерних архітектур, і фактичним обладнанням. Загалом програмне забезпечення включає спільні бібліотеки та операційну систему, оскільки вони можуть бути створені для

різних цільових системних архітектур. У комп'ютерній системі зазвичай існує чотири інтерфейси, за допомогою яких програмне забезпечення ініціює взаємодію з апаратним забезпеченням. Ці інтерфейси є архітектурою набору інструкцій (ISA), що взаємодіє між ОС і апаратним забезпеченням; інтерфейс ISA між загальними бібліотеками, програмним забезпеченням і обладнанням; прикладний бінарний інтерфейс (ABI) і інтерфейс прикладного програмування (API). На рисунку 2.8 показано ці 4 інтерфейси.

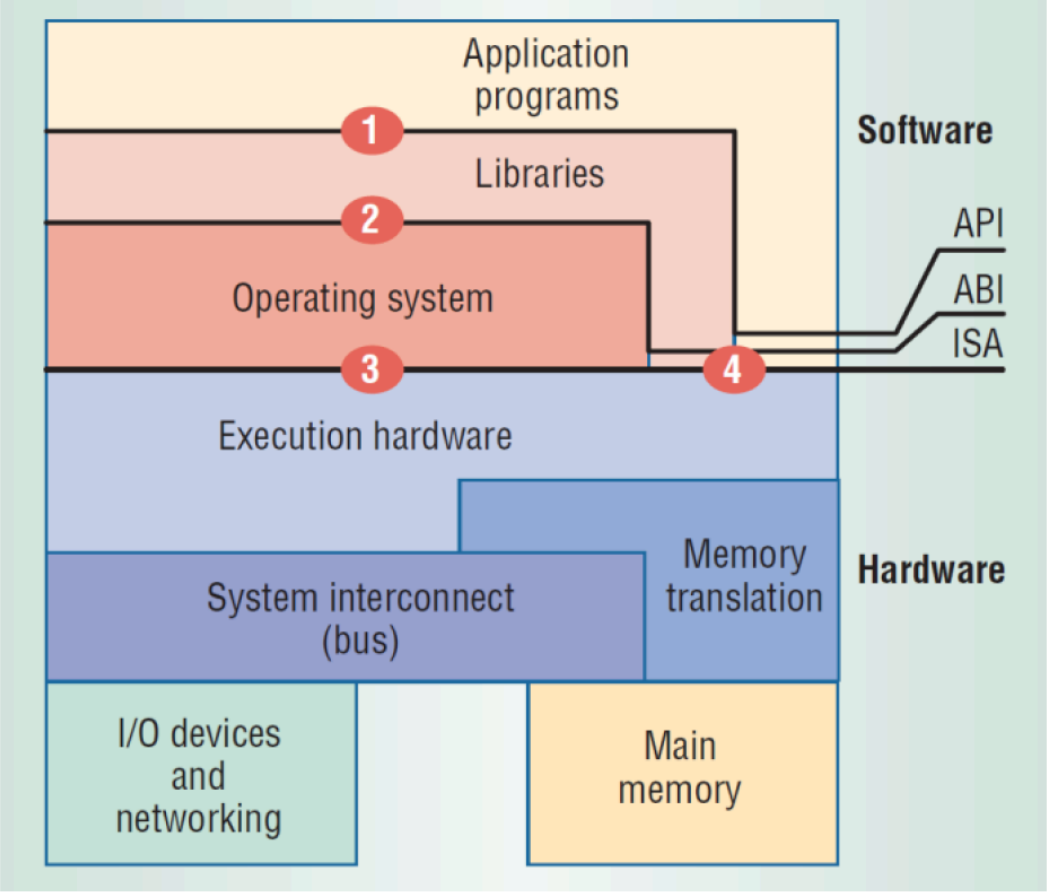


Рисунок 2.8 – Архітектура комп'ютерної системи та програмно-апаратні інтерфейси, які може реалізувати гіпервізор

Архітектура набору інструкцій (ISA) є основним інтерфейсом між будь-яким програмним забезпеченням і обладнанням. Він включає низькорівневі нативні інструкції, типи даних, розташування реєстрів, пов'язані з апаратним забезпеченням. Доступ до ISA в основному

здійснюється через операційну систему (через ядро), однак певне програмне забезпечення та спільні бібліотеки (наприклад, драйвери) також можуть мати інтерфейс, подібний до апаратного забезпечення. Це представлено інтерфейсом 4 та інтерфейсом 3 відповідно на схемі вище (рис. 2.7).

Двійковий інтерфейс програми (ABI) пропонує інтерфейс для скомпільованого програмного забезпечення та спільних бібліотек для доступу до обладнання через системні виклики, які є резидентними в операційній системі. Таким чином системні виклики з ОС діють як проксі фактичного набору інструкцій.

Інтерфейс прикладного програмування (API) забезпечує інтерфейс програмного забезпечення через спільні бібліотеки та системні бібліотеки, щоб отримати доступ до апаратних ресурсів. Інтерфейс API дозволив перекомпілювати програмне забезпечення для інших системних архітектур, за умови, що вже скомпільовані спільні бібліотеки та системні бібліотеки існують і мають той самий API.

Гіпервізор повинен мати можливість реалізувати один або декілька з цих інтерфейсів таким чином, щоб він міг розуміти виклики з вищих рівнів і модифікувати їх відповідно до фактичної архітектури фізичного обладнання нижче. Зворотний процес повернення апаратної відповіді програмному забезпеченню належним чином також є обов’язком гіпервізора.

Віртуалізація реалізується різними способами. Відмінності в способі його реалізації впливають на вимоги гостьової операційної системи та на вимоги конкретного гіпервізора, на якому він буде працювати. Нижче наведено різні реалізації віртуалізації.

Повна віртуалізація передбачає, що гіпервізор бере на себе повну відповідальність за віртуалізацію майже всіх можливостей, можливих через різні інтерфейси. Зазвичай це означає, що гостьові операційні системи не потребують жодних модифікацій для роботи на даному гіпервізорі. Недоліком цього є те, що гіпервізор може стати великим і громіздким інтерфейсом

реалізації, який може працювати з різними операційними системами та драйверами для периферійних пристроїв.

Паравіртуалізація — це тип віртуалізації, де гіпервізор має лише обмежений набір функцій серед інтерфейсів для підтримки гостьових операційних систем. Це означає, що гостьову операційну систему необхідно змінити, щоб вона працювала на даному гіпервізори. У більшості випадків гіпервізор надає спеціальний API, який можна використовувати для керування модифікацією операційної системи. Недоліком цього є те, що код ОС має бути доступним, а гостьові операційні системи потрібно перекомпілювати. Драйвери для цих операційних систем також необхідно переписати, щоб відповідати API гіпервізора. Перевагою цього є те, що гіпервізор залишається якомога меншим і ефективним. Крім того, модифікація операційної системи, ймовірно, проводиться експертами, які добре знають систему, щоб вбудувати будь-які оптимізації, які можуть знадобитися.

Висновки до розділу 2

В даному розділі виконано дослідження методів та підходів обробки даних в розподілених системах та хмарних інфраструктурах. Представлено опис методик отримання інформації з цифрових джерел, розподілених систем як вид хмарних цифрових рішень та стратегії розміщення ресурсів і механізми взаємодії. Наведені характеристики великомасштабних розподілених системи зберігання даних, мобільних та хмарних середовищ, наведені процеси віртуалізації та гіпервізори в хмарних інфраструктурах.

РОЗДІЛ 3. РОЗРОБКА СЕМАНТИЧНОЇ МОДЕЛІ ТА АРХІТЕКТУРИ ХМАРНОГО РІШЕННЯ РЕПОЗИТОРІЇВ ЦИФРОВИХ ДАНИХ

3.1 Використання семантичних веб-технологій для проектування хмарних систем

Семантична павутина спочатку була описана як метод, що дозволяє автоматизовано міркувати над даними, щоб дозволити машинним агентам обмінюватися, «розуміти» та обробляти різноманітні форми даних із різних джерел. таким чином, щоб допомогти користувачам. Початкова ідея полягала в тому, щоб розмітити текстові дані в Інтернеті значущою інформацією, яку могли б обробляти машини, керуючись попередньо встановленими значеннями певних типів даних, а також правилами, що стосуються зв'язків між типами даних. Замість того, щоб машини були розумними самі по собі, дані також могли б допомогти їм зробити висновки та забезпечити більш ефективну обробку. Це, у свою чергу, може допомогти людям автоматизувати та координувати діяльність за допомогою комп'ютера в більш раціональний та поінформований спосіб.

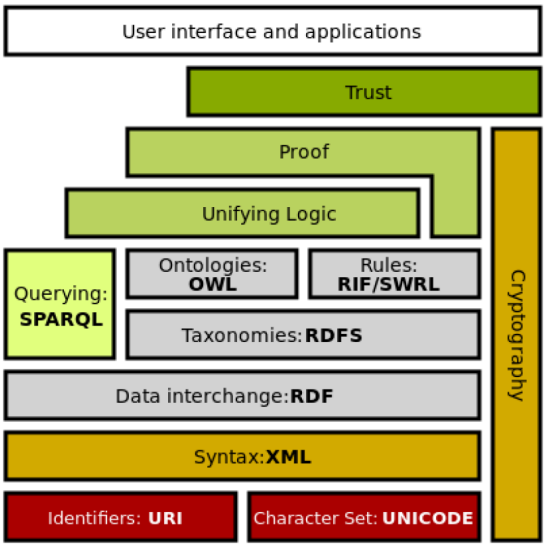


Рисунок 3.1 – Стек семантичного вебу / багатошарова модель

Технології, які складають семантичну мережу, все ще розвиваються, однак вони зазвичай включені в багаторівневу модель. На рисунку 3.1 зображено адаптацію моделі яку називають стеком семантичного вебу або багатошаровою моделлю.

3.3.1. Стек семантичної мережі

Стек семантичної мережі часто розкладають на 7 рівнів. Ці шари коротко описано нижче від нижнього рівня до верхнього:

1. Ідентифікатори та набори символів (URI та Unicode).Цей рівень утворює основу семантичної мережі для узгодженого обміну даними та адресації ресурсів. Для того, щоб машини могли інтерпретувати та обробляти дані, вони зазвичай виражаються в числовій формі – як правило, у вигляді двійкових цифр (бітів). Переклад між цією цифровою формою та текстовими символами (які складають основу людської письмової мови) може відрізнятися. Таким чином, щоб стандартизувати цю інтерпретацію між різними пристроями, використовуються набори символів. Семантична павутина використовує набір символів Unicode. Інші можливі набори символів включають ASCII та EBCDIC, однак вони тут не використовуються.

Уніфікований ідентифікатор ресурсу (URI) — це компактна послідовність символів, яка ідентифікує абстрактний або фізичний ресурс. Він надає прості, розширювані засоби для ідентифікації ресурсу, а також глобальну угоду про іменування, яка має глобальний обсяг і, таким чином, може інтерпретуватися послідовно в різних контекстах.

2. Синтаксис і простори імен (XML і XML-схема).Цей рівень анотує основні сутності (ресурси чи об’єкти) простими додатковими даними про сутність. Це відповідає синтаксису розширюваної мови розмітки (XML), який робить дані не лише машиночитаними, але й зрозумілими для людей. Відмінною рисою XML є те, що він розширюваний, оскільки він дозволяє довільне додавання тегів, які дозволяють анотувати нові дані. Схеми XML —

це шаблони, які дозволяють створювати конкретні класи на основі типів даних («наборів тегів») і обмежень. Такі XML-документи даних можна перевірити на відповідність їхнім відповідним схемам, щоб перевірити їх відповідність схемі, таким чином також забезпечуючи стандартизацію.

Простір імен XML дозволяє прив’язувати певні імена елементів або атрибути до конкретного домену використання. Це дозволяє уникнути конфліктів під час вирішення імені елемента чи атрибута, яке випадково має те саме ім’я, що й інший елемент чи атрибут у різних доменах. Таким чином, тут простір імен можна розглядати як аналогічний ідентифікації домену, в якому певний елемент або атрибут розуміється певним чином.

3. Опис сутності та прості зв’язки (RDF та RDF-схема). Цей рівень забезпечує прості зв’язки між сутностями, а також властивості, якими сутності можуть спільно користуватися через зв’язки. Таким чином самі дані можуть виражати прості значення через зв’язки, такі підмножини сутностей, подібно до концепції успадкування в об’єктно-орієнтованому програмуванні.

Структура опису ресурсів (RDF) виражає прості зв’язки між однією сутністю та іншою або сутністю з певною властивістю через концепцію «трійок». Трійка - це набір, що складається з суб'єкта, предиката та об'єкта. Суб'єкт завжди є сутністю. Предикат описує відношення, а об’єкт може бути сутністю або властивістю. Таким чином прості зв’язки між об’єктами даних можуть бути зв’язані в ланцюг, дозволяючи зробити простий висновок про інші зв’язки. Також можуть бути реалізовані ієрархії спадкування властивостей. Графічний приклад наведено нижче на рисунку 3.2, взятому з [35]. Об’єкт з URI, що містить ідентифікатор працівника «id132», має властивість «Jim Lerner» і зв’язок «has name». У той же час цей самий об’єкт даних з URI «id132» має зв’язок «authorOf» з іншим об’єктом даних з URI, що містить ідентифікатор ISBN «0012515866», який також має зв’язок «hasPrice» із властивістю «62 долари». Таким чином можна зробити висновок, що Jim Lerner є автором книги, яка коштує 62 долари.

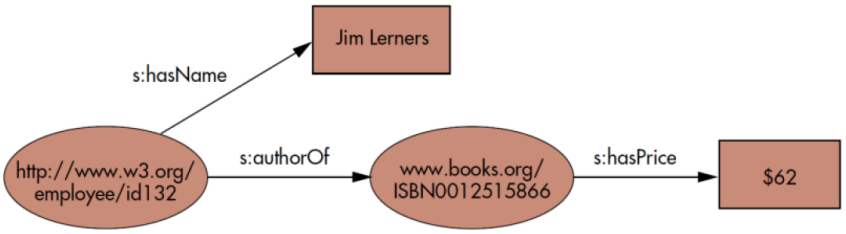


Рисунок 3.2 – Приклад RDF концепцій суб’єкта, предиката та об’єкта

Подібно до XML, який має схему XML для визначення структури та простої форми домену певних визначених типів даних, RDF також має схему RDF для створення простих словників, що містять правила для групування окремих класів сутностей і певних діапазонів властивостей або зв’язків. між сутностями та властивостями.

4. Онтологічний словник і правила (OWL). Цей рівень забезпечує додаткову виразність шляхом захоплення синтаксису вираження сутностей (термінів) і властивостей, а також зв’язків і правил, за допомогою яких можна застосовувати міркування щодо сутностей і властивостей. У семантичній мережі онтологія описується як «документ або файл, який формально описує зв’язки між термінами». Він має таксономію, яка визначає класи та підкласи, а також відносини між ними. Точніше, онтологія описує використання конкретного набору термінів щодо конкретної області через розробку його зв’язків з іншими термінами та правилами, які регулюють набір термінів, про який йде мова.

Хоча модель RDF також описує зв’язки, її зв’язки обмежуються посиланнями на властивості та зв’язками. Словники онтології, на додаток до опису таксономічних зв’язків, пропонують додаткову виразність у тому, що можуть описувати складні зв’язки, такі як відносини об’єднання та непересічні (членства), відносини рівності, обмеження значення властивості, обмеження властивості класу та обмеження кардинальності серед інших. Це дозволяє ще більше збагатити значущі зв’язки між термінами. Мова веб-онтологій (OWL) — стандарт, який широко використовується для опису

онтологій семантичної мережі. На додаток до цього, щоб охопити зв’язки, які, можливо, неможливо виразити безпосередньо в словнику онтології, були розроблені мови правил, такі як Semantic Web Rule Language (SWRL), які працюють у тандемі з мовами опису онтології, такими як OWL. Мови правил допомагають виводити нові зв’язки, не визначені явно в словнику. У випадку, якщо використовується кілька мов правил і правила потрібно перекласти для сумісності, був розроблений формат обміну правилами (RIF).

Також у сфері шарів представлення знань з використанням RDF і OWL була реалізована потреба в пошуку інформації або пошуку нових знань серед цих описів. Таким чином був розроблений простий протокол і мова запитів RDF (SPARQL). SPARQL пропонує SQL-подібний механізм запитів, який дозволяє повертати запити на певні набори даних і відповіді. SPARQL аналізує запити до графоподібної структури, яка є результатом посилань, створених на основі зв’язків у потрійках RDF. Розробка можливостей SPARQL все ще триває для забезпечення запитів із розподілених кінцевих точок із різними онтологіями. Наступні 3 рівні семантичної мережі все ще перебувають на ранній стадії розробки та ще не стандартизовані, однак їх призначені функціональні можливості були описані.

5. Уніфікована логічна структура. Цей рівень призначений для використання формальних логічних приписів і правил для більш складних висновків, які призводять до нових знань. Він може використовувати просту формальну логіку або більш складні концепції штучного інтелекту з математичними та статистичними основами, такими як байєсівський висновок.

6. Доказ. Доказ у семантичній мережі передбачає визначення правильності відповіді шляхом перевірки того, як вона була отримана (тобто логіка), перевірки використаних даних і перевірки довіри до ланцюжка постачальників даних. На цьому рівні також слід враховувати обмін доказами від різних агентів.

7. Довіра та криптографія (цифрові підписи та шифрування). Концепція довіри дуже пов’язана з процесом перевірки доказів. Цей рівень призначений для забезпечення того, щоб обмін інформацією та джерелами інформації можна було перевірити, перевірити та не змінити, щоб не зберігати помилкові чи спотворені знання. Довіра часто вимірюється за допомогою механізмів на основі репутації, механізмів на основі контексту або механізмів на основі контенту. Механізми, засновані на репутації, працюють на оцінці багатьох інших суб’єктів. Механізми, засновані на контексті, працюють шляхом оцінювання фонової інформації або створення метаданих під час генерування оцінюваного вмісту. Механізми, засновані на вмісті, оцінюють самі дані. Довіра також підвищується завдяки використанню криптографії, зокрема під час використання цифрових підписів. Цифрові підписи зазвичай постачаються у формі хеш-функцій, які реалізують дайджести повідомлень або коди автентифікації повідомлень на основі ключів. Мета цифрових підписів — уможливити перевірку автентичності даних, а також автентичності автора даних. Також може використовуватися використання інфраструктур відкритих ключів для перевірки вмісту повідомлення та його авторів за допомогою сертифікатів і третіх сторін. Слід зазначити, що нижні 4 рівні стеку Semantic Web пройшли процес стандартизації з рекомендаціями щодо дизайну та інструментами, але все ще вдосконалюються. Верхні 3 шари ще не стандартизовані та все ще відкрито обговорюються.

3.2 Розробка вимог та стандартів до розподіленої системи репозиторіїв цифрових даних

3.2.1. Основні стандарти

Відзначається, що сьогодні цифрові пристрої використовуються практично в кожному аспекті людської діяльності як на користь, так і на

шкоду суспільству. Цифрові пристрої можуть використовуватися як інструменти або об’єкти зловмисної діяльності. Зосереджуючись на необхідності розслідування цифрових злочинів, описано кілька постійних проблем, з якими зараз стикаються цифрові розслідування в сучасному мережевому середовищі.

Зокрема, кількість пристроїв, які обмінюються даними, неоднорідність цих пристроїв, складність між взаємодіями та швидкість взаємодії призводять до великої кількості даних, які потрібно відсіювати під час цифрових розслідувань.

Типова ємність зберігання та обчислення повсякденних цифрових пристроїв, таких як мобільні телефони, різко зростає. Крім того, зростає кількість пристроїв, які вилучаються як частина доказів у кримінальних справах. Різноманітність типів можливих цифрових джерел доказів зростає (наприклад, хмарне сховище, жива пам’ять, смартфони, GPS-пристрої, набори RAID, мережевий трафік), а також протоколів і схем кодування, що використовуються серед цих пристроїв. Загальний ефект цього полягає в тому, що в будь-якому конкретному цифровому розслідуванні необхідно обробити багато цифрових доказів, а правоохоронні органи стикаються з труднощами через нестачу навченого персоналу, брак обладнання та величезний тягар терабайтів даних, які потрібно просіяти.

Поточний набір галузевих стандартних інструментів, які використовуються для полегшення процесів цифрових розслідувань, також відсутній. Вони недостатньо оснащені для роботи з надзвичайно великими обсягами даних, а процедури аналізу виконуються переважно вручну. Лише такі дії, як індексування та пошук, є частково автоматизованими. Крім того, вони часто орієнтовані на докази або створені для вирішення конкретного завдання в конкретній ситуації, тому вони не здатні ефективно співвідносити численні різнорідні джерела доказів, а також їх нелегко розширювати. Крім того, багато з них обмежені у своїх можливостях проводити дистанційні розслідування в реальному часі та в основному використовують власні

формати, які не дозволяють обмінюватися доказами між інструментами та слідчими групами.

Загальний ефект полягає в тому, що цифрові розслідування сповільнюються.

На основі цих проблем, визначених у цьому документі, ми описуємо універсальну розподілену архітектуру, яка спрямована на прискорення процесу цифрового розслідування шляхом автоматизації та використання потужності пристроїв кінцевих точок користувача. Сама архітектура базується на передумові мережевого середовища, де є пристрої, які мають деякі резервні ресурси для сприяння процесу цифрового розслідування. Ми працюємо з припущенням, що мережеве середовище не обов'язково має бути під контролем корпоративного підприємства і може бути загальнодоступною мережею.

3.2.2. Вимоги до системи

Вимоги до такої розподіленої цифрової архітектури розглянемо дальше:

- **Розповсюдження:** можливість працювати з кількома користувачами, кількома джерелами даних, масштабованим сховищем, обробкою та розповсюдженням:
  - Багатокористувацький: можливість розподіляти завдання між кількома користувачами та зводити результати.
  - Масштабований: ресурси з точки зору зберігання, обробки та передачі повинні бути в змозі обробляти як великі, так і малі навантаження еластичним чином, отримуючи ресурси та вивільняючи їх відповідно до потреб.

Це також передбачає розширюваність функціональних можливостей для виконання нових завдань і здатність приєднуватися до вузлів-учасників і виходити з них.

- Доступність: ресурси та функціональні можливості повинні бути присутніми, коли це необхідно.
- Спільне використання ресурсів: об’єднання ресурсів із пристроїв-учасників для надання більшої ємності ресурсів, зокрема для допомоги пристроям з обмеженими ресурсами.

- **Універсальність:** робота з неоднорідністю та відсутністю стандартизації серед пристроїв і джерел доказів, що забезпечує співпрацю та інтеграцію аналітичних результатів:
  - Формати даних: можливість представлення різних форматів даних доказів в одному форматі, який може дозволити інтеграцію, спільний доступ і обробку в уніфікований спосіб кількома пристроями
  - Платформа: надання можливості пристроям із різними платформами та конфігураціями також брати участь у процесі. Дозволити, щоб вибір платформи для людини не був перешкодою.
- **Швидкість реагування**: миттєва реакція на інциденти безпеки та забезпечення інтерактивності, що дозволяє спільно інформований обмін інформацією. Цьому можна сприяти завдяки живій та віддаленій інтерактивності в мережевих середовищах:
  - Он-лайн (завжди на зв’язку): можливість взаємодії з цільовими пристроями під час інцидентів безпеки, що забезпечує швидший час реакції.
  - Віддалене: використання мережевого підключення для забезпечення віддаленого доступу до цільових пристроїв для зменшення затримки між виявленням інциденту безпеки та вилученням доказів.
- **Цілісність** (довіра, надійність і точність): підтримка необхідних рівнів достовірності та автентичності даних і процесу, а отже, і результатів:
  - Даних: хешування джерел даних цифрових доказів і перевірка сторін, які мали до них доступ, сприяють достовірності даних, гарантуючи уникнення необґрунтованих змін.

- Процес: документування завдань, залучених осіб (організацій) і оцінка суті фактичних процедур шляхом експертної оцінки сприяють достовірності процесів.

- **Конфіденційність:** забезпечення збереження конфіденційної та ідентифікаційної інформації в зашифрованому форматі, відсутність неправомірного використання та заборону неавторизованого доступу з метою сприяння прийняттю такої системи.

- **Безпека**: підтримка автентифікації, авторизації, підзвітності та неспростування вжитих заходів для забезпечення достовірності результатів:

  - Автентифікація: весь зв’язок має бути перевірений, щоб походити та завершуватися в організації, яка претендує на зазначену особу.

  - Авторизація: можливості мають бути обмежені лише тими, які потребують певних функцій.

  - Підзвітність: усі дії повинні передбачати ретроспективний аудит.

3.3. Розробка архітектури масштабованої однорангової розподіленої системи

Архітектура агрегатора інформації на основі гіпервізора для агрегування та аналізу цифрових доказів розроблена з урахуванням вищезазначених вимог.

Передбачається, що він складається з 4-рівневої архітектури:

а) гіпервізор на основі хоста (HbH);

б) архітектура однорангового розподілу (P2P-da);

в) хмарний бекенд (CBB);

г) контролер безпеки (LEC).

Нижче на рис. 3.3 і 3.4 наведено схематичне представлення системи, що зображують взаємодію між компонентами та ієрархію рівнів відповідно.

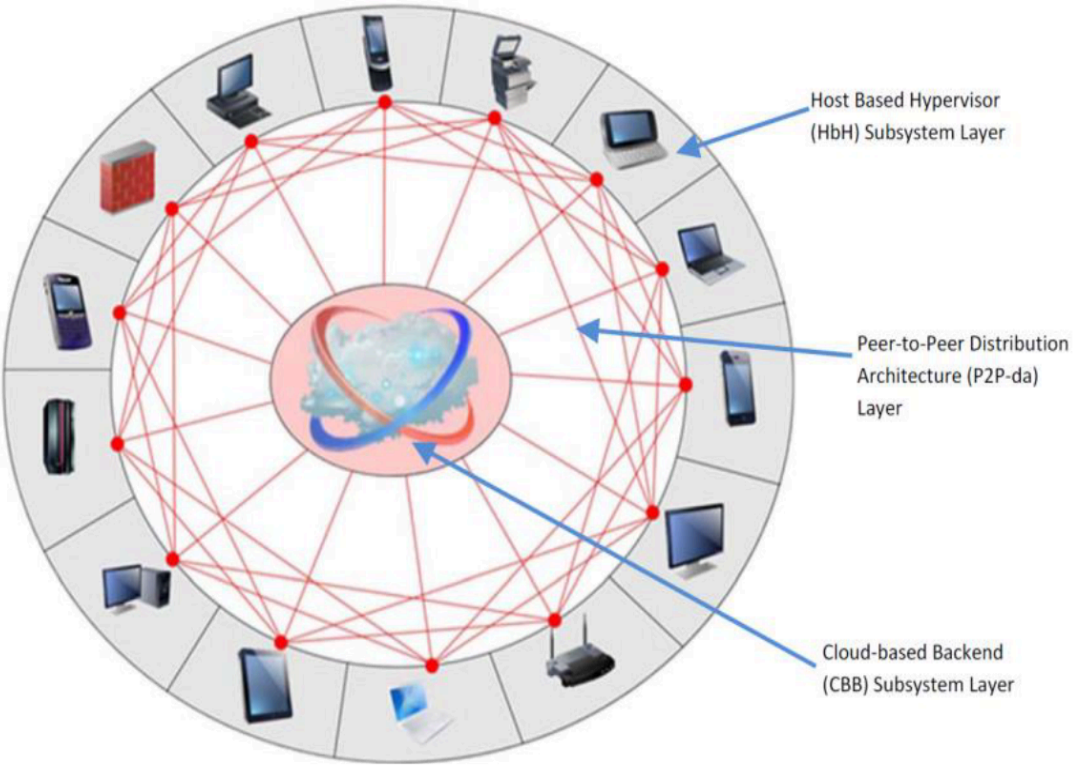


Рисунок 3.3 – Взаємодія між 3 рівнями архітектури пропонованого агрегатора інформації

Функціональність і компоненти кожного з шарів коротко пояснюються нижче:

- Гіпервізор на основі хоста (HbH). Компонент HbH складається з чистого гіпервізора, який забезпечує рівень віртуалізації на кожному пристрої, що бере участь. Як більш привілейований рівень програмного забезпечення, він має прямий доступ до апаратних інтерфейсів, а також до програмного забезпечення гостьової операційної системи. Таким чином, це може дозволити отримання цифрових доказів безпосередньо з джерела, а також увімкнути самоаналіз віртуальної машини для збору артефактів доказів. Вибір гіпервізора як основного пункту збору доказів сприяє цілісності та універсальності (як даних, так і платформи). Джерела доказів збираються та зберігаються у форматі AFF4, який спочатку базується на RDF-анотаціях концепцій доказів. Це дозволяє інтегрувати докази в хмарну серверну систему, що сприяє універсальності форматів доказів.

Мінімальна система виявлення вторгнень на основі хоста також вбудована в гіпервізор, щоб забезпечити швидке виявлення зловмисної активності в системі. Це допомагає швидше реагувати, дозволяючи виявлення в джерелі. Інформація про зловмисну активність передається іншим системам HbH, а також серверній частині на основі хмари. Це сприяє спільному використанню ресурсів щодо зібраних даних про загрози. Також зберігаються списки, що містять хеші відомих файлів, щоб зменшити розмір отриманих даних.

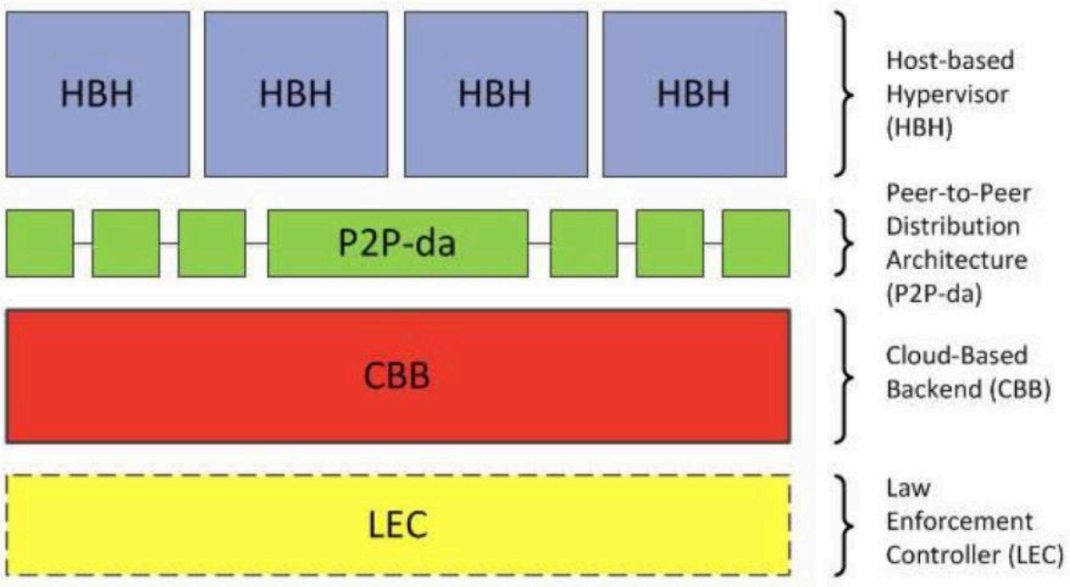


Рисунок 3.4 – Чотирьохрівнева модель архітектури пропонованого агрегатора інформації

- Архітектура одноранговогo розподілу (P2P-da). У поєднанні з системами HbH цей компонент служить для передачі інформації між окремими системами HbH, що беруть участь, і серверною частиною на основі хмари. P2P-da передає три типи інформації:

1) інформацію про можливу шкідливу діяльність розподіляється між пристроями HbH;

2) керівну інформацію щодо підтримки накладення P2P, а також команд, що видають оновлення для систем або реакції на порушення безпеки;

3) джерела отриманих даних (доказів) у формі дампів диска, дампів пам’яті або записів мережевого трафіку, які передаються до серверної частини на основі хмари.

З точки зору збору доказів, суть P2P-da полягає в тому, щоб передавати фрагменти доказів на пристрої з більшою кількістю ресурсів, щоб врешті-решт досягти серверної частини на основі хмари.

Метою архітектури P2P-da є можливість розповсюдження через масштабованість, доступність і спільне використання ресурсів, а також швидкість реагування через підключення до мережі. Масштабованість, доступність і спільне використання ресурсів є загальними характеристиками систем на основі P2P. Завдяки використанню епідемічних накладень масштабованість і доступність досягаються завдяки резервуванню та стійкості до відтоку, які вони забезпечують. Спільне використання ресурсів найкраще демонструється за допомогою градієнтних накладень, які організовують мережу таким чином, щоб завдання та апаратні ресурси можна було спільно використовувати з більш стабільними та потужними вузлами. Швидкість реагування досягається завдяки підключенню до мережі, оскільки воно забезпечує живий і віддалений доступ до цільових пристроїв. Як побічний продукт, безпека даних під час передачі та учасників за допомогою використання криптографії також сприяє вимогам цілісності, автентифікації, конфіденційності та, отже, конфіденційності.

- Хмарний бекенд (CBB). Ця підсистема складається з масштабованої розподіленої файлової системи, механізму керування відомими файлами та набору систем HbH із високим ресурсом. Масштабована розподілена файлова система у формі кластера розподіленої файлової системи Hadoop (HDFS) забезпечує масштабований і високодоступний постійний простір для зберігання, тим самим сприяючи вимогам масштабованості та доступності системи. Цілісність також є побічним продуктом цього вибору, оскільки HDFS підтримує цілісність даних між вузлами, автоматично ініціюючи дублювання елементів даних, втрачених

через пошкодження, наприклад, через збій диска. Механізм розрізнення керує відомими даними (відомими файлами, пам’яттю процесів або мережевими пакетами) таким чином, що раніше переглянуті дані не потрібно отримувати та передавати повністю для зберігання. Головні однорангові вузли HbH побудовані з високою потужністю ресурсів, щоб діяти як супер-однорангові вузли, які стабілізують загальне накладання однорангового вузла. Таким чином, вони сприяють виконанню вимог щодо спільного використання ресурсів у тому, що, будучи більш стабільними з вищою пропускною здатністю, більшою робочою пам’яттю та постійним сховищем, вони можуть спільно використовувати свої ресурси та протистояти попиту менш обладнаних пристроїв.

- Контролер безпеки (LEC). Ця підсистема призначена для забезпечення інтерфейсу співробітників щодо взаємодії з наборами даних доказів. Оскільки всі вони зберігаються в одному місці в єдиному форматі, їх можна інтегрувати, співвідносити, обґрунтовувати та запитувати, щоб зібрати більше інформації про поточне цифрове розслідування, уможливлюючи більш автоматизований аналіз цифрових доказів. Це забезпечує швидке реагування, оскільки дозволяє автоматизований аналіз із результатами, близькими до реального часу. Це також полегшує платформу для кількох користувачів, щоб аналізувати той самий набір доказових даних різними способами. Крім того, це сприяє універсальності даних, оскільки використовується єдине джерело даних, тоді як учасники, які використовують дані, можуть сильно відрізнятися.

Таким чином, коли ми глибше вивчаємо внутрішню роботу архітектури пропонованого агрегатора то більше зосереджуємося на методі віддаленого збору даних, використовуючи накладення P2P як метод автоматизації етапу збору даних. Ми більш детально описуємо внутрішню роботу архітектури однорангового розподілу (P2Pda) і додатково мотивуємо її необхідність через оцінку життєздатності еквівалента клієнт-сервер з точки зору його швидкості

та надійності як на смартфонах з обмеженими ресурсами, так і на смартфонах з більші ресурсні можливості.

3.4 Розробка структури хмарної системи на основі архітектури розподілу P2P-da

Основна мета P2P-da — забезпечити надійність, масштабованість і швидку пропускну здатність даних навіть в умовах високого рівня «відтоку» (вузли приєднуються до мережі та залишають її). Через P2P-da передається кілька форм даних. Це включає: інформацію про події безпеки (сповіщення), повідомлення (команди) про контроль інцидентів безпеки, повідомлення про обслуговування накладання та отримані дані цифрових доказів. Щоб досягти необхідної масштабованості, надійності та швидкої пропускної здатності, об’єднано кілька P2P-накладень (включаючи градієнтні накладення, епідемічні накладення та протокол BitTorrent). Є 3 основні види діяльності, які виконує P2P-da, і вони здійснюються за допомогою вищезгаданих накладень P2P:

1. Обслуговування накладення P2P полегшується за допомогою градієнтних і епідемічних накладень;
2. Поширення інформації про зловмисну діяльність і попередження полегшується через епідемічні накладки;
3. Збір даних реагування на інциденти полегшується завдяки адаптації протоколу BitTorrent. У цьому підрозділі більш детально розглядаються підтримка накладення P2P і збір даних про реагування на інциденти. Короткий виклад деталей наведено нижче.

3.4.1. Обслуговування мережі P2P

Основна мета цієї діяльності полягає в підтримці зв’язку між залученими вузлами в межах відповідного оверлею. Градієнтне накладання

має на меті підтримувати зв’язок між вузлами, які мають обмежені ресурси, з тими, які наділені більшою кількістю ресурсів і, отже, мають кращу стабільність. Це робиться за допомогою метрики корисності [42], яка відстежує кілька факторів на вузлі, включаючи доступні ресурси, поточне завантаження та час роботи вузла.

Для накладення кожному вузлу необхідно підтримувати знання про «поруч» і деяких «далеких» з точки зору метрики корисності. Це необхідно для того, щоб уникнути розбиття мережі через надмірну кластеризацію подібних вузлів.

Таким чином, кожен одноранговий вузол підтримує невеликий список інформації про адреси однорангових вузлів, які мають подібні показники корисності, і інший список інформації про адресування однорангових вузлів, які мають значно відмінні показники корисності.

За допомогою градієнта кожен одноранговий вузол повинен підтримувати невеликий список інформації про адреси однорангових вузлів, які мають менші показники корисності, і ще один список однорангових вузлів, які мають більші показники корисності.

Таким чином одноранговий вузол має контакт як з вузлами, які мають менше обчислювальних ресурсів, так і з тими, які краще забезпечені ресурсами.

Підмножини періодично обмінюються випадковим чином, тобто одноранговий вузол вибирає випадкового однорангового вузла з одного зі своїх списків і обмінюється випадковою підмножиною своїх списків з цим одноранговим вузлом. Необхідно підтримувати список «Нещодавно побачених однорангових пристроїв», щоб запобігти переключенню між одним і тим же набором однорангових пристроїв. Необхідно встановити порогові значення, щоб гарантувати, що однаковий набір даних не поширюється. Розміри кожного підтримуваного списку також потрібно оптимізувати.

3.4.2. Збір даних реагування на дії

Ця діяльність ініціюється, коли за допомогою комбінованого рішення між вбудованим IDS HbH і серверною частиною CBB визначається, чи необхідно виконувати збір доказів на конкретному вузлі. Ми не обговорюємо цей процес прийняття рішень. Ми обговорюємо передачу даних, отриманих із цільового пристрою, як адаптацію протоколу BitTorrent, що показано на рис. 3.5 як трифазний процес.

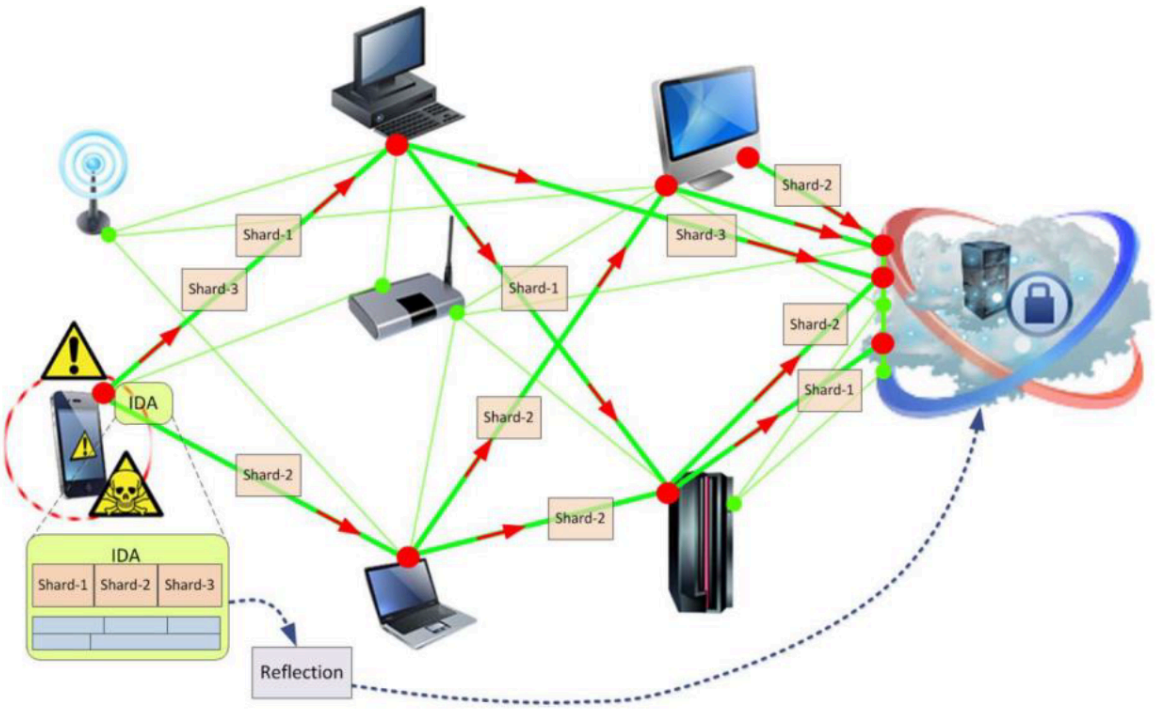


Рисунок 3.5 – Процес збору даних та реагування на дії

Розбиття даних. Після визначення джерела доказів (диск, пам’ять, мережеве захоплення або журнали) робиться відповідна копія джерела. Ця копія стискається та зберігається на пристрої у форматі RDFannotated AFF4 як архів даних інциденту, а потім розділяється на менші частини, які називаються шардами, підписаними та зашифрованими. Для цього створюється файл метаданих, що містить хеші MD5 і SHA1, і надсилається безпосередньо до CBB. Таким чином, згідно з протоколом BitTorrent, CBB діє

як «трекер» і «витягувач» з еквівалентом «торрент-файлу», який шукає інших однорангових пристроїв, які можуть мати необхідні сегменти архіву даних про інциденти.

Шардовий розподіл. Під час передачі шардів до СВВ градієнтне накладання полегшує розповсюдження кількох копій кожного шарда до кількох більш потужних однорангових пристроїв. Ці однорангові пристрої називаються «прихильниками» певного сегмента. Кожного разу, коли шард передається іншому піру, він підвищує свій «рівень». Після досягнення певного порогу система HbH зобов'язана безпосередньо завантажувати до головних партнерів HbH СВВ. Якщо це не вдається, виконується процедура виборів серед однорангових пристроїв, які мають шард, щоб визначити, яка система HbH завантажуватиме безпосередньо. Щоб уникнути постійного вибору одного однорангового вузла, «значення залежності» встановлюється таким чином, що одноранговий вузол може виконувати лише певну кількість прямих завантажень протягом певного періоду часу.

Швидка реконструкція фрагментів. Кожен шард потенційно завантажується з кількох вузлів HbH, які мають певний сегмент. Це робиться для забезпечення доступності, а також для спільного використання пропускної здатності. Подібно до протоколу BitTorrent, пріоритет надається найрідкіснішим фрагментам, тобто тим, які мають найменшу кількість прихильників. Коли шарди надходять у СВВ, вони перевіряються на цілісність і зберігаються. Якщо шард є помилковим, ініціюються завантаження від інших підтримок. Після успішного завантаження фрагменти видаляються з інших вузлів.

3.5 Побудова онтологічних моделей та семантичне представлення процесів інтеграції цифрових даних

Описано п'яти-етапний метод, який демонструє процес перетворення необроблених цифрових даних у формат, який можна легко інтегрувати в інші

інструменти, а також має невід’ємну здатність автоматичного обґрунтування та інтелектуального запиту. Цей метод передбачає: збір (отримання) даних, семантичне представлення, онтологічне міркування, міркування на основі правил та інтегровані запити. Нижче наведено діаграму (рис. 3.6), що зображує процес.

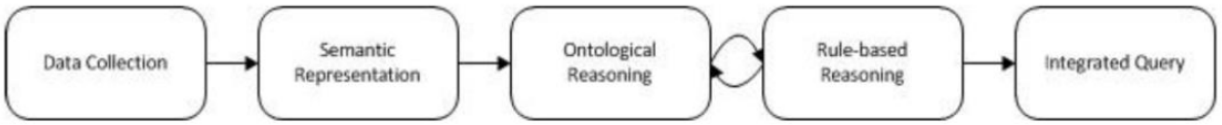


Рисунок 3.6 – Метод семантичного представлення та інтеграції даних

Нижче коротко описано дії, які беруть участь у кожному етапі методу:

1. Збір даних: це передбачає отримання джерела цифрових даних (наприклад, образ диска, дампів пам’яті або записів мережевого трафіку) у спосіб і виконання методів зменшення даних, таких як видалення відомих файлів, якщо необхідно

2. Семантичне представлення: тут необроблені дані аналізуються та перетворюються на свою семантичну модель, представлену твердженнями RDF, що з’єднують онтологічні сутності та властивості відповідно до вибраної онтології, що відповідає мові, такій як OWL. Отримані набори тверджень від кількох аналізаторів зберігаються в одному файлі (починаючи процес інтеграції).

3. Онтологічне міркування: цей крок передбачає подачу семантичного представлення в механізм міркування OWL, який виводить нові твердження (відносини) на основі вхідного набору даних і заданих онтологій. Таким чином реалізуються нові прості зв’язки між об’єктами, які повертаються в семантичне представлення, яке потрібно зберегти.

4. Обґрунтування на основі правил: це можна розглядати як іншу фазу або як підфазу попередньої фази. Оскільки онтологічних міркувань може бути недостатньо для виявлення більш складних відносин, мови правил можна використовувати для встановлення зв’язків між об’єктами

(індивідуумами). Це особливо корисно для зв’язування об’єктів із різнорідних джерел. Слід зазначити, що онтологічне міркування та фази міркування на основі правил виконуються ітеративно, а не послідовно, оскільки онтологічні висновки можуть допомогти сформувати аксіоми/основи, корисні для тверджень на основі правил, і може статися навпаки, так що твердження на основі правил може створювати зв’язки для аксіом/основ для онтологічних висновків.

5. Інтегрований запит: на цьому кроці результуючі твердження RDF, які були обґрунтовані за допомогою як онтологічних міркувань, так і міркувань на основі правил, утворюють єдину масу пов’язаних об’єктів, що нагадують граф. Цей графік представляє зв’язки між доказовими артефактами з кількох джерел, однак інтегрованими в одну модель. Пошук у цій моделі здійснюється за допомогою SPARQL, який дозволяє здійснювати пошук у графах RDF подібно до того, як працює SQL. Таким чином, слідчий може здійснити комплексний уніфікований пошук по всій сукупності даних з багатьох джерел.

Інтеграція тверджень RDF може відбуватися або через процес онтологічного міркування, або через процес міркування на основі правил. У цьому дослідженні ми встановлюємо 3 прості форми інтеграції, 2 які виникають через просте онтологічне міркування, а 1 відбувається за допомогою реалізації правила:

1. Інтеграція індивідів в одному просторі імен і в одній доменній онтології. Наприклад: якщо IP-адреса з’являється кілька разів в одному джерелі доказів і використовується та сама онтологія. Це може бути реалізовано всередині онтології та виконано через онтологічне міркування.

2. Інтеграція індивідів у різних просторах імен, але в одній онтології домену. Наприклад, коли IP-адреса з’являється в різних джерелах доказів, але використовується та сама онтологія домену. У цьому випадку унікальна ідентифікаційна властивість може бути використана для встановлення

зв’язку, а онтологічне міркування може встановити, що вони є однією особою.

3. Інтеграція індивідів у різних просторах імен і різних онтологіях домену. Наприклад, якщо файл завантажується та зберігається на диску, той самий файловий об’єкт може з’являтися у 2 різних наборах доказів і визначатися як 2 різні особи. Для того, щоб встановити, що вони є однією і тією ж особою, необхідно використовувати правило. Таким чином, необхідно застосовувати міркування на основі правил, оскільки онтології домену можуть бути описані по-різному. Ми використовуємо SPARQL у нашому підході для визначення правил і, зокрема, визначаємо правило для пов’язування файлів, знайдених у мережевому трафіку, та файлів, знайдених у файловій системі, як однієї особи через їхній хеш MD5.

Щоб продемонструвати цей метод, ми встановили експеримент для моделювання порушення безпеки, яке призвело б до наявності дискових і мережевих джерел доказів, які ми могли б отримати. Зокрема, ми встановили «drive-bydownload», який передбачає завантаження та виконання шкідливого файлу на диск і його виконання. Джерелами доказів були образ диска та мережевий трафік. Наскільки нам було відомо, не існувало добре відомих онтологій, які б описували концепції цифрових даних, тому ми створили полегшені онтології для артефактів доказів дискового зберігання та артефактів доказів мережевого трафіку. Їх частини показано на малюнках 3.7 і 3.8.

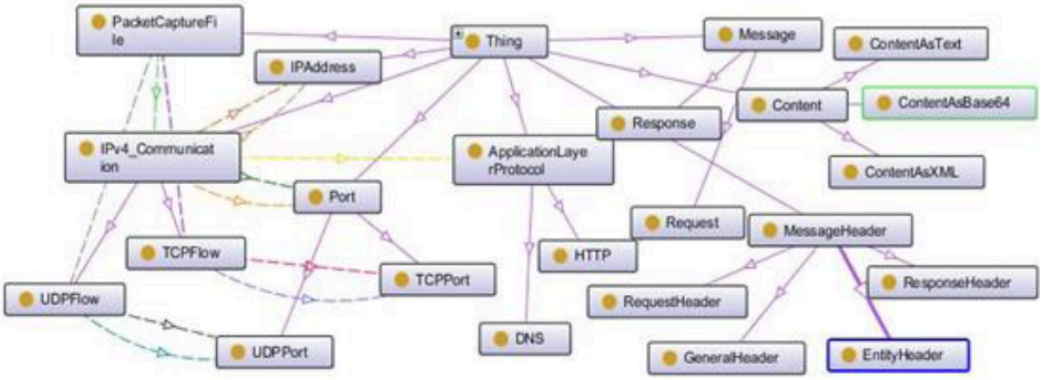


Рисунок 3.7 – Онтологічна модель концепцій захоплення мережевих пакетів

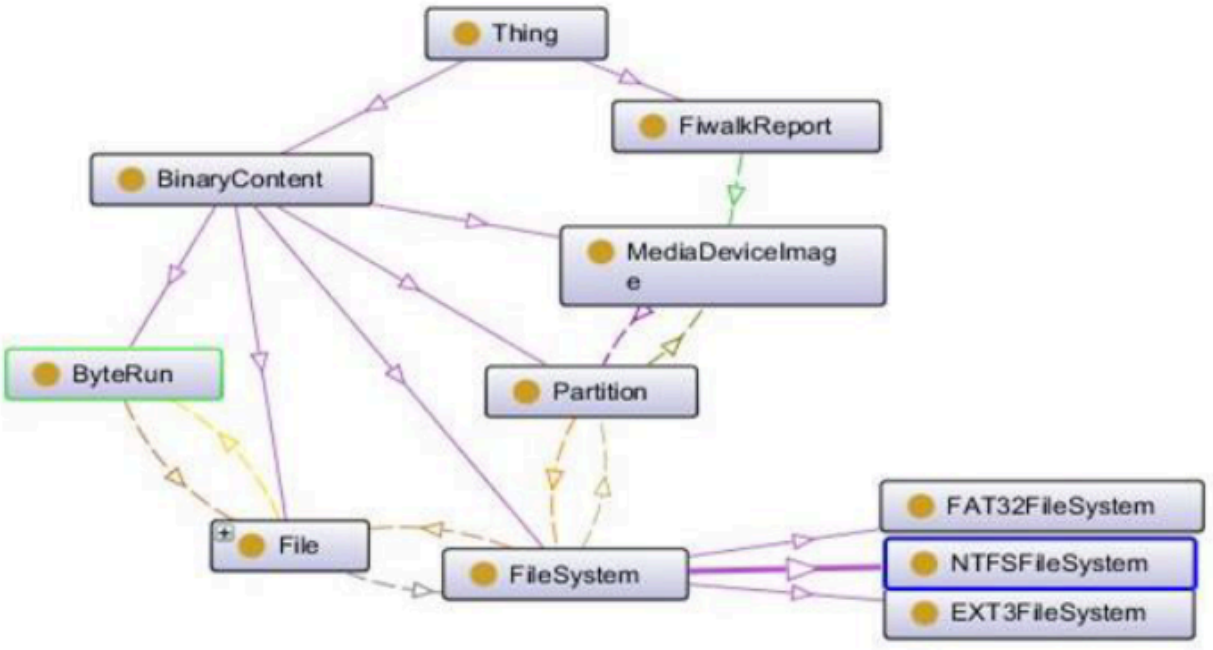


Рисунок 3.8 – Онтологічна модель концепцій файлової системи

Ми також створили аналізатори, які перетворюють образи дисків і мережеві захоплення у відповідні твердження RDF і визначення онтології та об’єднують їх в один файл, що представляє великий графік тверджень RDF. Отриманий єдиний граф тверджень RDF було подано в механізм міркування OWL і механізм міркування на основі правил разом із спрощеними онтологіями та правилами, розробленими раніше.

Онтологічні міркування та процес міркування на основі правил автоматизовані за допомогою механізму міркування Pellet і механізму правил SPARQL.

За допомогою механізму SPARQL для виконання конкретних запитів, таких як «Які файли з однаковим вмістом завантажено з Інтернету та знайдено в кількох системах?», ми можемо встановити з результату, що процес міркування інтеграції артефактів доказів із різнорідних джерел дійсно відбулося.

Таким чином можна надсилати інші запити системі SPARQL і автоматично отримувати обґрунтовані відповіді на основі висновків щодо інтегрованих наборів даних з цифрового репозитарію.

Висновки до розділу 3

Отже, в цьому розділі представлено процес розробки семантичної моделі та архітектури хмарних цифрових рішень, із використанням семантичних веб-технологій та стеку семантичної мережі. Здійснена розробка вимог та стандартів до розподіленої системи репозиторіїв цифрових даних, представлена пропонована архітектура масштабованої однорангової розподіленої системи.

ВИСНОВКИ

В поданій кваліфікаційній роботі виконана оптимізація моделей та засобів реалізації хмарних розподілених систем репозиторіїв цифрових даних шляхом побудови архітектури агрегатора інформації на основі гіпервізора для агрегування та аналізу даних. Виконано визначення вимог до системи виконання автоматизованого аналізу цифрових даних для широкомасштабного загальнодоступного мережевого сценарію, такого як той, що спостерігається в мобільних і хмарних середовищах.

Запропонована архітектура, що складається з 4 рівнів: гіпервізор на основі хоста (HbH), архітектура однорангового розподілу (P2P-da), хмарна архітектура бекенда (CBB) і контролера безпеки. Використання гіпервізора в компоненті архітектури Host-based Hypervisor (HbH) усуває неоднорідність різноманітних програмних платформ, що дозволяє створити єдину точку збору цифрових даних. Крім того, гіпервізор сприяє довірі до процесу отримання даних і цілісності джерел даних, оскільки він забезпечує більш привілейовану платформу, з якої можна отримати цифрові дані, ніж за допомогою розміщеного програмного агента.

В роботі розкрита суть 2P-da яка полягає в тому, щоб уможливити віддалений збір даних, а також обмін інформацією про певні події між вузлами мережі. Швидкість реагування полегшується завдяки обміну інформацією про інциденти, а також завдяки використанню градієнтного накладання, яке фокусується на організації мережі таким чином, щоб пристрої з обмеженими ресурсами завжди мали шляхи зв’язку з вузлами, які забезпечені краще. Градієнтне накладання також допомагає досягти кращого спільного використання ресурсів, забезпечуючи цей зв’язок між пристроями з обмеженими ресурсами та їх більш забезпеченими та стабільнішими аналогами.

Досліджено процес розробки семантичної моделі та архітектури хмарних цифрових рішень, наведений стек семантичних веб-технологій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Alink, W., Bhoedjang, R.A.F., Boncz, P.A., de Vries, A.P., XIRAF - XML-based indexing and querying for digital forensics. Digital Investigation 3, 50–58. 2006.

2. Almulhem, A., Traore, I., Experience with Engineering a Network Forensics System, Proceedings of the 2005 international conference on Information Networking. Convergence in Broadband and Mobile Networking. Springer Berlin Heidelberg, Korea. 2005.

3. Andrus, J., Dall, C., Hof, A. Van, Laadan, O., Nieh, J., Cells : A Virtual Mobile Smartphone Architecture. SOSP ’11 Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles 173–187. 2011.

4. Armstrong, A.K., Mueller, J.J., Syrett, T., The Smartphone Royalty Stack: Surveying Royalty Demands for the Components Within Modern Smartphones. SSRN Electronic Journal 116, 1–69. 2014.

5. Attiya, H., Welch, J., Distributed Computing - Fundamentials, Simulations and Advanced Topics, 2nd ed. John Wiley & Sons Inc. Ayers, D., 2009. A second generation computer forensic analysis system. Digital Investigation 6, S34–S42. 2009.

6. Bader, M., Baggili, I., iPhone 3GS Forensics : Logical analysis using Apple iTunes Backup Utility. Small Scale Digital Device Forensics Journal 4, 1–15. 2010.

7. Beebe, N., Digital forensic research: The good, the bad and the unaddressed. Advances in digital forensics V 17–36. 2009.

8. Beebe, N.L., Clark, J.G., A hierarchical, objectives-based framework for the digital investigations process. Digital Investigation 2, 147–167. 2005.

9. Bejtlich, R., The Practice of Network Security Monitoring: Understanding Incident Detection and Response. No Starch Press. 2013.

10. Berners-Lee, T., Fielding, R., Masinter, L., RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax. Network Working Group. 2005.

11. Berners-Lee, T., Hendler, J., Larissa, O., The Semantic Web. Scientific American 284, 34–43. 2001.

12. Bhoedjang, R.A.F. a F., van Ballegooij, A.R.R., Van Beek, H.M.A.M. a, van Schie, J.C.C., Dillema, F.W.W., Van Baar, R.B.B., Ouwendijk, F.A. a., Streppel, M., 2012.

13. Birrell, A.D., Nelson, B.J., Implementing Remote procedure calls. ACM SIGOPS Operating Systems Review 17, 3. 1983.

14. Bishop, M., Introduction to Computer Security. Prentice Hall. Borman, D., Braden, B., Jacobson, V., Scheffenegger, R., 2014. RFC 7323 TCP Extensions for High Performance, RFC 7323. 2005.

15. Bray, T., Paoli, J., Sperberg-McQueen, M., Extensible Markup Language (XML) 1.0 : W3C Recommendation 10-Feb-98, Language. 1998.

16. Breeuwsma, M., Jongh, M. De, Forensic data recovery from flash memory. Small Scale Digital … 1, 1–17. 2007.

17. Breeuwsma, M.F., Forensic imaging of embedded systems using JTAG (boundary-scan). Digital Investigation 3, 32–42. 2006.

18. Bress, S., Menz, M.J.,. Write protection for computer long-term memory devices 2. 2004

19. Brezinski, D., Killalea, T., RFC3227 - Guidelines for Evidence Collection and Archiving (No. 55), Best Current Practice. 2002.

20. Cafarella, M., Cutting, D., Apache HBase [WWW Document]. Apache Software Foundation. URL https://hbase.apache.org/ 2007.

21. Carrier, B., Spafford, E., Getting physical with the digital investigation process. International Journal of Digital Evidence 2, 1–20. 2003.

22. Carrier, B.D., Spafford, E.H., An event-based digital forensic investigation framework. In: Digital Forensics Research Workshop. pp. 1–12. 2004.

23. Carroll, J., Labs, H., Bizer, C., Berlin, F.U., Hayes, P., Stickler, P., The Sematic Web Trust Layer. New York 1–17. 2004.

24. Casadei, F., Forensics and SIM cards : an Overview. International Journal 5, 1–21. 2006.

25. Case, A., Cristina, A., Marziale, L., Richard, G.G., Roussev, V., FACE: Automated digital evidence discovery and correlation. Digital Investigation 5, S65–S75. 2008.

26. Casey, E., What does “forensically sound” really mean? Digital Investigation 4, 49–50. 2007.

27. Casey, E., Digital Evidence and Computer Crime: Forensic Science, Computers and the Internet, Third. ed. Elsevier Inc. 2011. Casey, E., 2013. Triage in digital forensics. Digital Investigation 10, 85–86.

28. Casey, E., Altheide, C., Daywalt, C., Donno, A. de, Forte, D., Holley, J.O., Johnston, A., Knijff, R. van der, Kokocinski, A., Luehr, P.H., Maguire, T., Pittman, R.D., Rose, C.W., Schwerha IV, J.J., Shaver, D., Smith, J.R., Handbook of Digital Forensics and Investigation, Handbook of digital forensics and investigation. 2010.

29. Casey, E., Back, G., Barnum, S., Leveraging CybOXTM to standardize representation and exchange of digital forensic information. Digital Investigation 12, S102–S110. 2015.

30. Casey, E., Ferraro, M., Nguyen, L., Investigation delayed is justice denied: proposals for expediting forensic examinations of digital evidence. Journal of forensic sciences 54, 1353–64. 2009.

31. Casey, E., Stanley, A., Tool review - Remote forensic preservation and examination tools. Digital Investigation 1, 284–297. 2004.

32. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E., Bigtable: A distributed storage system for structured data. In: 7th Symposium on Operating Systems Design and Implementation (OSDI ’06), November 6-8, Seattle, WA, USA. pp. 205–218. 2006.

33. Chockler, G., Melamed, R., Tock, Y., Vitenberg, R., SpiderCast: A Scalable Interest-Aware Overlay for Topic-Based Pub/Sub Communication. Proceedings of the 2007 Inaugural International Conference on Distributed Event-Based Systems (DEBS) 14–25. 2007.

34. Chung, H., Park, J., Lee, S., Kang, C., Digital forensic investigation of cloud storage services. Digital Investigation 9, 81–95. 2012.

35. Cisco Systems Inc., Cisco Global Cloud Index: Forecast and Methodology, 2014–2019, Cisco Global Cloud Index. 2015.

36. Clemens, J., Automatic classification of object code using machine learning. Digital Investigation 14, S156–S162. 2015.

37. Cohen, B Incentives build robustness in BitTorrent. Workshop on Economics of Peer-to-Peer systems. 2003.

38. Cohen, M., Garfinkel, S., Schatz, B., Extending the advanced forensic format to accommodate multiple data sources, logical evidence, arbitrary information and forensic workflow. Digital Investigation 6, S57–S68. 2009.

39. Cohen, M., Schatz, B., Hash based disk imaging using AFF4. Digital Investigation 7, S121–S128. 2010.

40. Cohen, M.I., Advanced carving techniques. Digital Investigation 4, 119–128. 2007.

41. Cohen, M.I., PyFlag – An advanced network forensic framework. Digital Investigation 5, S112–S120. 2008.

42. Cohen, M.I., Bilby, D., Caronni, G., Distributed forensics and incident response in the enterprise. In: Digital Investigation. Elsevier Ltd, pp. S101–S110. 2011.

43. Connolly, J., Davidson, M., Schmidt, C., The Trusted Automated eXchange of Indicator Information (TAXII TM). 2014.

44. Coulouris, G., Dollimore, J., Kindberg, T., Distributed Systems: Concepts and Design, Computer. 2012.

45. Cruz, F., Moser, A., Cohen, M., A scalable file based data store for forensic analysis. Digital Investigation 12, S90–S101. 2015.

метадані

Заголовок

Оптимізація моделей та засобів реалізації хмарних розподілених систем репозиторіїв цифрових даних

Автор

Куруляк О.О. Науковий керівник / Експерт

підрозділ

King Danylo University

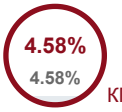
Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

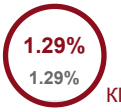
Заміна букв	ß	1
Інтервали	A→	0
Мікропробіли	·	4
Білі знаки	ß	0
Парафрази (SmartMarks)	a	68

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1



КП 2

25
Довжина фрази для коефіцієнта подібності 2

17993
Кількість слів

136935
Кількість символів

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	http://repository.ukd.edu.ua/bitstream/handle/123456789/391/%D0%9F%D0%B0%D1%85%D0%BE%D0%BB%D1%8C%D1%87%D1%83%D0%BA%20%D0%9E.%D0%A0.%20%D0%B4%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC%D0%BD%D0%B0.pdf?sequence=1	44	0.24 %
2	http://repository.ukd.edu.ua/bitstream/handle/123456789/395/%D0%94%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0%20%D0%A1%D1%82%D0%B5%D0%BF%D0%B0%D0%BD%D1%8E%D0%BA.pdf?sequence=1	44	0.24 %
3	http://repository.ukd.edu.ua/bitstream/handle/123456789/391/%D0%9F%D0%B0%D1%85%D0%BE%D0%BB%D1%8C%D1%87%D1%83%D0%BA%20%D0%9E.%D0%A0.%20%D0%B4%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC%D0%BD%D0%B0.pdf?sequence=1	35	0.19 %