Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Метод створення відмовостійких багатошарових віртуальних інфраструктур в хмарному середовищі

(тема)

Виконав:
студент II курсу, групи СПм-20-1
Бондаренко М.Е.
(прізвище, ініціали)

Спеціальність
123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: доц. Ткачов В.М.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Бондаренку Максиму Едуардовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Метод створення відмовостійких багатошарових віртуальних інфраструкур в хмарному середовищі

затверджена наказом по університету від “ 05 ” листопада 2021 р. № 1657 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 13 грудня 2021 р.

3. Вхідні дані до роботи
IAAS
Віртуальні системи
Мережа

4. Перелік питань, що потрібно опрацювати в роботі
Аналіз предметної області
Методи сучасного захисту віртуальних інфраструктур в хмарному середовищі
Моделювання сценаріїв атак на інфратруктуру в хмарному середовищі

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 16 арк. ф. А4

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз завдання	09.11.2021-14.11.2021	
2	Аналіз науково-технічної літератури	15.11.2021-18.11.2021	
3	Пошук існуючих моделей	19.11.2021-22.11.2021	
4	Пошук аналогічних моделей	23.11.2021-25.11.2021	
5	Імітаційне моделювання	25.11.2021-28.11.2021	
6	Оформлення пояснювальної записки	28.11.2021-04.12.2021	
7	Оформлення графічної частини	04.12.2021-06.12.2021	

Дата видачі завдання 8 листопада 2021 р.

Студент _____
(підпис)

Керівник роботи _____ доц Ткачов В.М.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 80 с., 17 рис., 1 табл., 1 дод., 25 джерел.

SAAS, IAAS, ПЕРЕДАЧА ДАНИХ, ПРОГРАМНИЙ ЗАСІБ, ВІДМОВОСТІЙКІСТЬ, ВІРТУАЛЬНІ МАШИНИ, ХМАРИ

Метою кваліфікаційної роботи є підвищення захищеності від збоїв під час роботи шляхом створення відмовостійких багатошарових віртуальних інфраструктур в хмарному середовищі. У ході виконання валіфікаційної роботи проаналізовані вузькі місця віртуальної інфраструктури, зроблено аналіз існуючих рішень, визначено їх недоліки, запропоновано новий метод, що забезпечує відмовостійку роботу багатошарових віртуальних інфраструктур та здійснено злеседування роботи методу шляхом імітації виходу з ладу окремих її вузлів та можливість відновлення системи у разі мережевих атак. За результатами досліджень було зроблено висновки щодо використання методу на рівні промислової експлуатації.

ABSTRACT

Master’s thesis: 76 pages, 17 figures, 1 tables, 1 appendices, 25 sources.

SAAS, IAAS, DATA TRANSMISSION, SOFTWARE, FAILURE RESISTANCE, VIRTUAL MACHINES, CLOUDS

The major goal of this thesis is to increase protection against failures during operation by creating fault-tolerant multilayer virtual infrastructures in a cloud environment. During the attestation work the bottlenecks of the virtual infrastructure were analyzed, the analysis of existing solutions was made, their shortcomings were identified, a new method was proposed, which provides fault-tolerant operation of multilayer virtual infrastructures. network attacks. Based on the results of research, conclusions were made regarding the use of the method at the level of industrial exploitation.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ОЕМ – область ефективної маршрутизації .

ОКМ – оверлейні комп’ютерні мережі.

VPN – узагальнена назва технологій, які дозволяють створювати віртуальні захищені мережі поверх інших мереж із меншим рівнем довіри.

DEVops – низка практик, призначених для пожвавлення взаємодії розробників із фахівцями інформаційно-технологічного обслуговування та зближення їхніх робочих процесів одне з одним.

ОС – операційна система.

ПК – персональний комп’ютер.

ЦП – центральний процесор.

ПЗ – програмне забезпечення.

MS – Microsoft Server.

СКБД – набір взаємопов’язаних даних (база даних) і програм для доступу до цих даних.

SLA – угода між постачальником послуг і користувачем про рівень послуг. Містить кількісні та якісні характеристики наданих послуг, такі як їх доступність, підтримка користувачів, час виправлення несправності та інше.

IaaS – це модель обслуговування, в межах якої споживачу надається можливість керувати засобами обробки та збереження, комунікаційними мережами, та іншими фундаментальними обчислювальними ресурсами, на базі яких споживач може розгортати та виконувати довільне програмне забезпечення, до складу якого можуть входити операційні системи та прикладні програми.

SaaS – модель поширення програм споживачам, при якій постачальник розробляє веб-програму, розміщує її й керує нею (самостійно або через третіх осіб) з метою використання її замовниками через інтернет.

PaaS – модель надання хмарних обчислень, при якій споживач отримує

доступ до використання інформаційно-технологічних платформ: операційних систем, систем управління базами даних, зв’язного програмного забезпечення, засобів розробки і тестування розміщених у хмарних провайдерах.

IT – це система методів, процесів та способів використання обчислювальної техніки і систем зв’язку для створення, збору, передачі, пошуку, оброблення та поширення інформації з метою ефективної організації діяльності людей.

API – набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

ITIL – описує найкращу світову практику організації підприємства чи підрозділу, що надає послуги у сфері інформаційних технологій.

NGFW – брандмауер нового покоління.

DPI – точок на дюйм.

NAT – це механізм у мережах TCP/IP, котрий дозволяє змінювати IP-адресу у заголовку пакунку, котрий проходить через пристрій маршрутизації трафіку.

OSI – абстрактна мережева модель для комунікацій і розробки мережевих протоколів. Представляє рівневий підхід до мережі.

LAN – локальна мережа.

IoT – концепція мережі, що складається із взаємозв’язаних фізичних пристроїв, які мають вбудовані давачі, а також програмне забезпечення, що дозволяє здійснювати передачу і обмін даними між фізичним світом і комп’ютерними системами в автоматичному режимі, за допомогою використання стандартних протоколів.

CI/CD – це комбінована практика безперервної інтеграції та безперервної доставки або безперервного розгортання. CI/CD доповнює прогалини між розробками та робочими групами, забезпечуючи автоматизацію при створенні, тестуванні та розгортанні програм.

DDoS – напад на комп’ютерну систему з наміром зробити комп’ютерні ресурси недоступними користувачам, для яких комп’ютерна система була

призначена.

ОЗУ – швидкодійна комп’ютерна пам’ять, призначена для запису, зберігання та читання інформації у процесі її обробки.

HTTP – протокол передачі даних, що використовується в комп’ютерних мережах. Назва скорочена від HyperText Transfer Protocol, протокол передачі гіпертекстових документів HTTP належить до протоколів моделі OSI 7-го прикладного рівні.

ВСТУП

У сучасному технологічному світі дедалі більшу нішу займають хмарні послуги. Представники бізнесу обирають дані рішення через їхню невисоку вартість та відсутність потреби в кадрах для конфігурації, налаштування та підтримки їхньої стабільної роботи. Також існуючі рішення мають різні концепції захисту даних для підвищення безпеки цілісності даних користувачів. Одним із таких методів є використання VPN-ланцюгів. Завдяки повторному пакету трафіку дані отримують високий рівень захисту і найголовніше, що кінцевий користувач не втрачає швидкодію системи.

На сьогоднішній день хмарні послуги стають ключовими для користувачів. Багато компаній змушені на годину пандемії змінити формат роботи співробітників, що тягне за собою технічні труднощі. Ідеальним рішенням для цього виступають хмарні платформи, на яких DEVops фахівці мають можливість у найкоротші терміни встановити всі потрібні ресурси та програмне забезпечення для використання великими командами зі своїх домашніх робочих станцій.

Головним питанням є стабільність та відмовостійкість системи, оскільки при збої роботи провайдера хмарних послуг бізнес-група зазнає великих збитків.

Тому на сьогоднішній день актуальним завданням є завдання створення відмовостійких віртуальних інфраструктур в хмарній середовищі за умови мінімального впливу на швидкодію системи в цілому.

Запропонований метод поєднує конфігураційне рішення на основі об’єднання методу балансування завантаження серверів та метод збереження стану обчислювальних процесів на основі "контрольних точок".

В валіфікаційній роботі поставлено завдання створення методу стійкості до відмови віртуальних інфраструктур у хмарній середовищі, а також провести аналіз впливу окремих її вузлів та можливість їх автоматичного відновлення.

1 ОСОБЛИВОСТІ АРХІТЕКТУРІ ВІРТУАЛІЗАЦІЇ З ВИКОРИСТАННЯМ ПОХУДНИХ ТЕХНОЛОГІЙ

1.1 Віртуалізація та хмарні послуги

Віртуалізація – це процес створення віртуального середовища на існуючому сервері для запуску бажаної програми без втручання в інші сервіси, які надає сервер або хост-платформа іншим користувачам. Віртуальне середовище може являти собою один примірник або комбінацію безлічі таких, як операційні системи, мережеві сервери або сервери додатків, обчислювальні середовища, пристрої зберігання та інші подібні середовища.

Слово «віртуалізація» в останній час стало «модою» в IT-середовищі. Всі сучасні вендори технічного оснащення та ПЗ, всі IT-компанії говорять, що віртуалізація – це сучасно, і потрібно всім користувачам вже сьогодні задуматися про їх застосування у своїх цілях.

Віртуалізація поділяється на три типи:

- віртуалізація уявлень;
- віртуалізація програм;
- віртуалізація серверів.

Численні переваги забезпечуються віртуалізацією, що включає зниження витрат, ефективне використання ресурсів, кращу доступність і мінімізацію ризиків серед інших [1].

Переваги для компаній:

- видалення спеціальних апаратних та службових вимог;
- ефективне керування ресурсами;
- підвищення продуктивності праці працівників у результаті кращої доступності;
- зниження ризику втрати даних, оскільки резервне копіювання даних здійснюється в кількох місцях зберігання;

- переваги для центрів обробки даних;
- максимізація можливостей сервера, тим самим знижуючи витрати на обслуговування та експлуатацію;
- менша займана площа внаслідок зниження вимог до обладнання, енергії та робочої потужності.

Найяскравіший приклад віртуалізації образів – це термінальні служби Windows Server. Термінальний сервер надає свої обчислювальні ресурси клієнтам, і клієнтський додаток виконується на сервері, клієнт же отримує лише «картинку», тобто уявлення. Така модель доступу дозволяє, по-перше – зменшити вимоги до програмно-апаратного забезпечення за клієнта, по-друге – знижує вимоги до пропускної здатності мережі, по-третє – дозволяє підвищити безпеку. Що стосується обладнання – то як термінальні клієнти можуть використовуватися навіть смартфони чи старі комп'ютери аж до Pentium 166, не говорячи вже про спеціалізованих тонких клієнтів. Існують, наприклад, тонкі клієнти у форм-факторі розетки Legrand, що монтуються у короб. На клієнтських робочих місцях достатньо встановити лише монітор, клавіатуру та мишу – і можна працювати, забезпечити високошвидкісне підключення до локальної мережі, навіть низькошвидкісного підключення з пропускною здатністю 15-20 кбіт/с, тому термінальні рішення дуже підходять фірмам, що мають сильно розподілену структуру (наприклад – мережі невеликих магазинів). Крім того, при використанні тонких клієнтів значно підвищується безпека, тому що користувачам можна дозволити запускати лише обмежений набір додатків та заборонити встановлювати свої власні додатки. Те саме можна зробити і з повноцінними клієнтськими робочими станціями, але з використанням термінальних служб це буде зробити набагато простіше, особливо – не надаючи доступу повністю до робочого столу, а лише публікуючи окремі програми (можливо в Citrix Metaframe/PS, а також Windows Server 2008 і вище). Більше того, жодну інформацію не можна буде скопіювати на та із зовнішнього носія, якщо це явно не дозволено при налаштуваннях термінальних служб. Тобто проблема «вірусів на флешках» відпадає автоматично. Ще одне безперечне

перевага – зниження складності адміністрування: спрощується оновлення додатків (достатньо оновити їх на сервері), та спрощується робота служб підтримки: до термінальної сесія будь-якого користувача можна підключитися віддалено без встановлення додаткового ПЗ. жодну інформацію не можна буде скопіювати на та із зовнішнього носія, якщо це явно не дозволено в налаштуваннях термінальних служб. Тобто проблема «вірусів на флеш-носіях» відпадає автоматично. Ще одне безперечне перевага – зниження складності адміністрування: спрощується оновлення додатків (досить обновити їх на сервері), та спрощується робота служб підтримки: до термінальної сесії будь-якого користувача можна підключитися віддалено без встановлення додаткового ПЗ. Жодну інформацію не можна буде скопіювати на та із зовнішнього носія, якщо це явно не дозволено в налаштуваннях термінальних служб. Тобто проблема «вірусів на флешках» відпадає автоматично. Ще одне безперечне перевага – зниження складності адміністрування: спрощується оновлення додатків (достатньо обновити їх на сервері), та спрощується робота служб підтримки: до термінальної сесія будь-якого користувача можна підключитися віддалено без встановлення додаткового ПЗ.

Недоліків таких систем два: по-перше – необхідність купівлі потужніших серверів (хоча це може бути дешевше, ніж безліч клієнтських робочих станцій з ТТХ, достатніми для запуску додатків локально), по-друге – поява єдиної точки відмови у вигляді термінального сервера. Ця проблема вирішується за рахунок використання кластерів, або ферм серверів, але це призводить до ще більшого подорожчання системи.

Віртуалізація програмних додатків – відносно новий напрямок. Віртуалізація програм дозволяє запускати окрему програму у своєму власному ізольованому середовищі (іноді називається «пісочниця», sandbox). Такий спосіб допомагає вирішити багато проблем. По-перше – знову ж таки безпека: додаток, запущений в ізольованому середовищі – не здатний нанести шкоди ОС та іншим додаткам. По-друге – всі віртуалізовані програми можна оновлювати централізовано з одного джерела. По-третє – віртуалізація додатків дозволяє

запускати одному фізичному ПК кілька різних додатків, що конфліктують один з одним, а також кілька різних версій однієї й тієї програми.

Віртуалізація серверів – це програмна імітація за допомогою спеціального програмного забезпечення апаратного забезпечення комп'ютера: процесор, пам'ять, жорсткий диск тощо. Далі, на такий віртуальний комп'ютер можна встановити операційну систему, і вона працюватиме на ньому так само, як і на простому, залізному комп'ютері. Найцікавіша перевага цієї технології – це можливість запуску кількох віртуальних комп'ютерів усередині одного «залізного», причому всі віртуальні комп'ютери можуть працювати незалежно один від одного [2].

Віртуалізацію серверів можна використовувати з метою навчання та тестових цілях. Наприклад, нову програму чи операційну систему можна протестувати у віртуальному середовищі, перш ніж її запустити в комерційну експлуатацію, без необхідності купувати «апаратне забезпечення» спеціально для цієї мети та без ризику паралічу IT-інфраструктури, коли щось піде не так. Однак, крім того, віртуалізація серверів може використовуватись у виробничій середовищі. Віртуалізація дозволяє зменшити кількість серверів за допомогою консолідації, тобто раніше потрібно було кілька серверів – тепер ви можете розмістити один сервер і запустити необхідне кількість гостьових операційних систем у віртуальному середовищі. Це дозволить заощадити витрати на придбання обладнання та зменшити енергоспоживання, тим самим зменшивши тепловіддачу системи, тож можна використовувати менш потужну, а отже, дешевшу систему охолодження. Але ця медаль має інший бік, Справа в тому, що впровадження рішень на основі віртуалізації, ймовірно, зажадає придбання нових серверів. Справа в тому, що віртуальні сервери використовують фізичні ресурси фізичних серверів і тому вимагають більш потужних процесорів, більше оперативної пам'яті та високошвидкісних дискових підсистем, і, ймовірно, більше. Крім того, деякі системи віртуалізації (включаючи MS Hyper-V) вимагають від ЦП підтримки технології апаратної віртуалізації (Intel VT або AMD-V) та деяких інших функцій ЦП. Багато процесорів, які були випущені

донедавна, особливо всі процесори x86_32-bit, не відповідають цим вимогам, тому від старих, хоча і повністю функціональних серверів доведеться відмовитися.

Є ще один дуже важливий момент: віртуалізація серверів дозволяє спростити адміністрування інфраструктури. Головне перевага, яку стосується усіх системних адміністраторів, – це можливість віддаленого доступу до консолі віртуальних серверів на «апаратному», точніше – «віртуально-апаратному» рівні, незалежно від встановленої гостьової ОС та її стану. Так щоб перезавантажити сервер, що вийшов зі строю, тепер не потрібно бігти в серверну, або купувати дороге обладнання типу IP-KVM-перемикачів, достатньо просто зайти в консоль віртуального сервера і натиснути кнопку «Reset». Крім цього, віртуальні сервери підтримують технологію миттєвих знімків, а також бекапи та відновлення віртуальних систем набагато легше.

Ще одне незаперечне перевага – ОС, запущена всередині віртуальної машини (хостова ОС), поняття не має, яке обладнання встановлено на фізичному сервері, всередині якого вона працює (хост). Тому при заміні заліза, при апгрейді або навіть переїзді на новий сервер необхідно обновити драйвери тільки на ОС самого хоста (хостової ОС). Гості ОС будуть працювати як і раніше, оскільки мають можливість комунікувати тільки віртуальні пристрої.

Також важливо, що у віртуальному середовищі можуть діяти особливі правила ліцензування ПЗ (зокрема, купівля ліцензії на Microsoft Windows Server 2008 Enterprise дозволяє використовувати безкоштовно чотири копії ОС як гостєву, а Microsoft Windows Server 2008 Datacenter взагалі дозволяє використовувати необмежену кількість хостових ОС за умови повного ліцензування за процесорами).

Важливим є технології відмовостійкості Фізичні сервери, на яких запускаються віртуальні машини, можуть бути об’єднані в кластер, і у разі відмови одного із серверів автоматично «переїжджати» на інший. Повної відмови стійкості домогтися не завжди можливо (зокрема, у MS Hyper-V такий «раптовий переїзд» виглядатиме так само, і мати такі ж можливі наслідки, як

раптове знеструмлення сервера), але можливі простої сильно скорочуватися: «переїзд» займає кілька хвилин, тоді як ремонт або заміна самого сервера може зайняти часи, а то й дні. Якщо ж переїзд віртуальних машин відбувається у штатному режимі, він може пройти абсолютно непомітно для користувачів. Такі технології у різних вендорів називаються по-різному, наприклад, у MS вона називається Live Migration, у VMware - Vmotion. Використання таких технологій дозволити проводити роботи. язані з вимкненням сервера (наприклад – заміну деяких апаратних компонентів, або перезавантаження ОС після встановлення критичних обновлень) у робочу годину і не виганяючи користувачів з улюблених додатків. Крім цього, якщо інфраструктура побудована відповідним чином – запущені віртуальні машини можуть автоматично переміщатися на менш навантажені сервери, або навпаки розвантажувати найбільш завантажені. В інфраструктурі на базі технологій Microsoft для цього використовують System Center Virtual Machine Manager та Operations Manager. або перезавантаження ОС після встановлення критичних обновлень) у робочий час і не виганяючи користувачів з улюблених програм. Крім цього, якщо інфраструктура побудована відповідним чином – запущені віртуальні машини можуть автоматично переміщатися на менш навантажені сервери, або навпаки «розвантажувати» найбільш завантажені. В інфраструктурі на базі технологій Microsoft для цього використовують System Center Virtual Machine Manager та Operations Manager. або перезавантаження ОС після встановлення критичних обновлень) у робочий час і не виганяючи користувачів з улюблених програм. Крім цього, якщо інфраструктура побудована відповідним чином – запущені віртуальні машини можуть автоматично переміщатися на менш навантажені сервери, або навпаки розвантажувати найбільш завантажені. В інфраструктурі на базі технологій Microsoft для цього використовують System Center Virtual Machine Manager та Operations Manager. або перезавантаження ОС після встановлення критичних обновлень) у робочий час і не виганяючи користувачів з улюблених програм. Крім цього, якщо інфраструктура побудована відповідним чином – запущені віртуальні машини можуть автоматично переміщатися на

менш навантажені сервери, або навпаки розвантажувати найбільш завантажені. В інфраструктурі на базі технологій Microsoft для цього використовують System Center Virtual Machine Manager та Operations Manager. або перезавантаження ОС після встановлення критичних обновлень) у робочий час і не виганяючи користувачів з улюблених програм. Крім цього, якщо інфраструктура побудована відповідним чином – запущені віртуальні машини можуть автоматично переміщатися на менш навантажені сервери, або навпаки розвантажувати найбільш завантажені. В інфраструктурі на базі технологій Microsoft для цього використовують System Center Virtual Machine Manager та Operations Manager.

Насамкінець теми з віртуалізації серверів – віртуалізація не завжди однаково корисна. Зокрема, не завжди буде гарною ідеєю переносити у віртуальну среду високонавантажені сервери, а особливо високонавантажені по дисковій підсистемі – це «важкі» СКБД, Exchange Server, особливо роль Mailbox Server та інші високонавантажені програми. А ось сервери із меншим навантаженням (контролери доменів AD, WSUS, всілякі System Center Manager, веб-сервера) віртуалізувати можна і навіть потрібно. Зауважу, до речі, що саме з контролерами доменів дуже бажано, щоб хоч один із контролерів був «залізним», тобто не віртуальним. Потрібно це тому, що для коректної роботи всієї інфраструктури бажано, щоб при запуску решти всіх серверів хоча б один контролер доменів уже був доступний у мережі [3].

Одним із популярним видом використання віртуалізації є використання хмарних сервісів (рисунок 1.1).

Хмарні послуги (cloud services) — це програми та платформи, які розміщуються та працюють на серверах хмарних провайдерів. Головне перевага хмарних програм полягає в тому, що створюючи обліковий запис на такій платформі, користувач має змогу отримувати доступ до власних файлів та інформації в будь-якому куточку планети, якщо має Інтернет-з’єднання. Потрібно лише придумати логін та пароль. Використовувати хмарні служби не лише зручно, а й безпечно. Навіть якщо з телефоном чи комп’ютером щось

трапиться – дані не зникнуть, їх можна буде відновити за декілька хвилин.

Аналогічна ситуація с сектором бізнеса та інших підприємств. Принцип роботи хмарних платформ досить простий. Користувач отримує хмарне сховище обраного розміру та позбавляється від необхідності купувати та обслуговувати власну інфраструктуру для збереження та керування даних. Це забезпечує гнучкість, швидку масштабованість та надійність.

Хмари поділяють за принципом поширення загальнодоступні (публічні), приватні та гібридні.

Публічна хмара – це величезна кількість доступних обчислювальних ресурсів, таких як мережа, пам’ять, центральний процесор та сховище. Ці ресурси розміщуються в одному з глобальнорозподілених і повністю керованих центрів обробки даних постачальника публічної хмари, і користувач може орендувати ці ресурси для створення IT-інфраструктури.

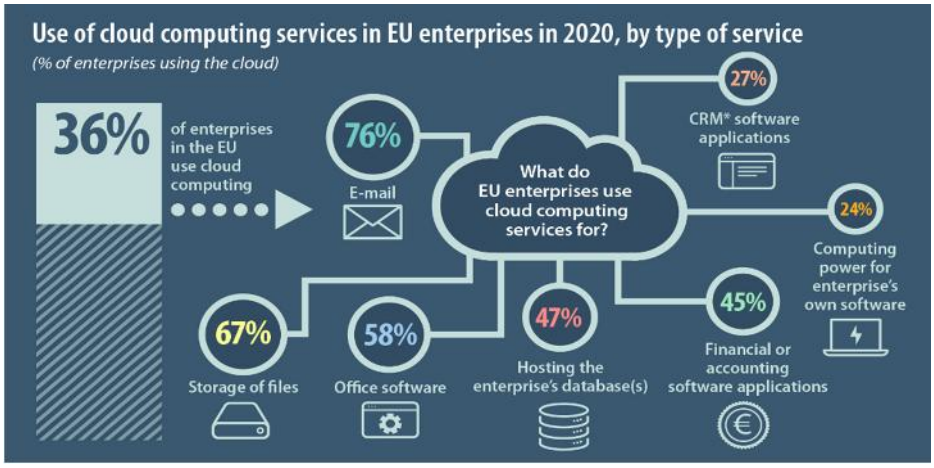

Рисунок 1.1 – Використання хмарних сервісів

Ці базові обчислювальні ресурси поєднуються з керованими послугами, такими як сервери баз даних, програми та системи безпеки. Керовані послуги можна орендувати, якщо вам не потрібні клопоти з налаштування та управління всім рішенням. Доступ до ресурсів у цьому типі хмар може бути простим, як використання веб-браузера.

Однією з головних переваг публічного хмари є те, що базове обладнання та логіка розміщуються, належать та обслуговуються кожним із цих

постачальників. Це означає, що клієнти не несуть відповідальності за купівлю або обслуговування фізичних компонентів, на яких складаються IT-рішення для публічної хмари [4].

Модель "оплата за фактом", яка використовується для оплати цих ресурсів, робить їх більш економічно ефективним рішенням, ніж самостійне володіння ними, оскільки користувачі платять лише за те, що споживаєте. Можливість масштабувати розмір рішення для задоволення піків та спадів у використанні заощаджує гроші клієнта, а також забезпечує величезну гнучкість.

Угоди про рівень обслуговування (SLA), підкріплені фінансовими ресурсами, зобов’язують кожного постачальника щомісяця забезпечувати процент безвідмовної роботи та гарантувати безпеку відповідно до таких стандартів, як GDPR, FIPS, HIPAPA та ін. Враховуючи це, постачальники громадських хмар інвестували та продовжують інвестувати десятки мільярдів фунтів у свої центри обробки даних, щоб забезпечити їх найсучаснішими відмовностійкими джерелами живлення, мережевими каналами, системами зберігання даних та автоматизованими системами моніторингу та обслуговування для виконання цих SLA.

Приватні хмари належать та використовуються окремими приватними підприємствами та організаціями. Зазвичай вони фізично розміщуються в центрі обробки даних підприємства за допомогою власного обладнання.

Однак підприємство може винайняти стороннього провайдера для розміщення приватної хмари на своїй площадці. У цьому сценарії приватна хмара має деяку подібність до публічного хмари, оскільки ресурси знаходяться у віддаленому керованому центрі обробки даних. Однак, хоча ці провайдери будуть пропонувати адміністративні послуги, вони зможуть запропонувати лише малу частку глобальних послуг публічного хмари.

Переваг цього типу і те, що й приватне хмара розміщується у власному центрі обробки даних, то власник отримує перевагу в тому, що може повністю контролювати всі рішення.

Оскільки має повний контроль за інфраструктурою, може адаптувати

хмарні обчислення до своїх переваг та внутрішніх процесів. Деякі з найсуворіших законів про безпеку та відповідність нормативним вимогам вимагають, щоб певні типи даних та ресурсів зберігалися в межах свого кордону безпеки – приватна хмара з власним хостингом допоможе дотриматися цієї вимоги.

Гібридна хмара – це комбінація елементів публічної та приватної хмари, які надійно з'єднані через Інтернет віртуальною приватною мережею (VPN) або виділеним приватним каналом.

Наприклад, користувач може використовувати практично необмежену ємність публічної хмари для зберігання даних, в той час як обробка даних може відбуватися на його території. Або може розширити свою комп'ютерну мережу до хмари, щоб не купувати додаткове стаціонарне обладнання.

Гібридне хмарне рішення пропонує переваги обох варіантів і уможливлює розширення хмари. У прикладі з розширенням приватної хмарної мережі це означає, що якщо закінчуються обчислювальні потужності в поміщенні, вони можуть бути надані із загальнодоступного хмари. Це економічно ефективний спосіб для підприємств збільшити обчислювальні потужності на вимогу, використовуючи при цьому вже оплачені ресурси в приміщенні.

Хмарна інфраструктура включає апаратні та програмні компоненти, необхідні для належної реалізації моделі хмарних обчислень. Хмарні обчислення також можна назвати обчисленнями, що надаються на вимогу.

Назва "хмарні обчислення" була навіяна символом хмари, яке часто використовується для представлення інтернету в блок-схемах та діаграмах.

Протягом багатьох років розробники програмного забезпечення та мережеві адміністратори використовували образ хмари, щоб уявити величезну кількість комунікаційних деталей, які виникають, коли при передачі повідомлень через Інтернет з однієї комп'ютерної мережі в іншу. Ця абстракція хмари тепер розширилася і включає процесори, як фізичні, так і віртуальні, сховища даних, рішення "програмне забезпечення як послуга" та мобільні додатки. Сьогодні хмарні програми та нові можливості з'являються щодня та приносячи з собою

нижчу вартість, моделі процесорів та сховищ даних з оплатою за використання моделі, більшу масштабованість, покращену продуктивність, простоту резервування та покращену безперервність бізнесу.

Хмарні обчислення працюють, дозволяючи клієнтським пристроям отримувати доступ до даних через Інтернет із віддалених серверів, баз даних та комп'ютерів.

Підключення до мережі Інтернет пов'язує передню частину (що включає клієнтське пристрій, браузер, мережу та хмарні програмні програми) із задньою частиною, що складається з баз даних, серверів та комп'ютерів.) Бек-енд функціонує як сховище, в якому зберігаються дані, доступ до яких отримує фронт-енд.

Зв'язок між лицевою та зворотною сторонами керується центральним сервером. Центральний сервер опирається на протоколи для полегшення обміну даними. Центральний сервер використовує програмне забезпечення та проміжне програмне забезпечення для управління зв'язком між різними клієнтськими пристроями та хмарними серверами. Як правило, для кожної окремої програми виділяється окремий сервер.

Парадигма створення хмарних інфраструктур в наш час пов'язана насамперед із використанням методології вкладеної віртуалізації. В основі цієї методології лежить низка принципів, моделей, методів, а також технологій, що їх реалізують.

Хмарні обчислення пропонують широкий спектр переваг для підприємств від зниження витрат до більш ефективного відновлення після аварій і підприємства підхоплюють цю тенденцію.

Форум хмарної індустрії (CIF) виявив, що 88% сучасних підприємств зараз використовують хмарні послуги, що на 83% більше від початку дослідження у 2010 році. Але після прийняття рішення щодо використання хмарних обчислень IT-менеджери повинні визначити, яку модель хмарних послуг використовувати: IaaS, PaaS або SaaS. Більшість сучасних компаній використовують комбінацію, оскільки кожна з них пропонує різні типи рішень.

Існує 3 основні типи хмарних обчислень як послуги, і кожен з них передбачає певну ступінь управління: інфраструктуру як послугу (IaaS), платформу як послугу (PaaS) та програмне забезпечення як послугу (SaaS) (рисунок1.2).

IaaS, або інфраструктура як послуга, працює аналогічно до традиційного комп'ютерного обладнання таким, як сервери, мережі, операційні системи, але функціонує у віртуальному режимі (рисунок 1.3). Замість того, щоб купувати фізичне обладнання, IT-менеджери можуть придбати інфраструктуру у вигляді віртуальної послуги через постачальника IaaS. Вони легко масштабуються та пропонують гнучкі хмарні обчислення, а також повний контроль за інфраструктурою вашої компанії. Прикладами IaaS є Amazon Web Services та Google Compute Engine.

Рисунок 1.2 – Рівень доступу хмарних інфраструктур

Перевагі IaaS інфраструктур:

- вартість. Хмарні програми, як правило, дешевші, тому що вони можуть повторно використовуватись і ділити ресурси. Не потрібно жодних попередніх інвестицій, і немає потреби у висококваліфікованому персоналі;

- масштабованість. Ресурсів єдиного сервера може не вистачати, але хмарні системи масштабуються дуже просто, і легко справляються з різкими змінами навантаження;

- надійність. Деяка надмірність хмарних систем означає, що вони суттєво надійніші за автономні системи;

- продуктивність. Хмарні послуги постійно моніторяться та покращують швидкодію;

- безпека. Хмарні системи володіють більшим рівнем безпеки, ніж їх автономні аналоги – і насамперед це зумовлено існуючою стурбованістю клієнтів цим питанням;

- доступ до можливостей інфраструктури найвищого рівня. Малий і середній бізнес може скористатися додатками та послугами, які спочатку створені для великих підприємств.

Недоліки IaaS інфраструктур:

- прив’язка до мережі Інтернет. Користувачі повинні мати постійне підключення (дротове та бездротове) до Інтернету для доступу до даних та програм;

- контроль даних. Малий бізнес залежить від постачальника хмарних послуг щодо захисту своїх конфіденційних даних;

- вимкнення. Найчастіше SLA не може перекрити ступінь руйнівного впливу відключень працювати малих підприємств. Навіть невеликий простий може обійтися величезними в масштабах малого бізнесу збитками;

- управління мобільними пристроями: користувач все ще стикається з проблемами безпеки з вашого боку, тому що користувачі продовжують використовувати особисті пристрої (ноутбуки, планшети та смартфони) для роботи через інтернет.

PaaS, або платформа як послуга, відноситься до хмарних сервісів, що надають основу, яку компанії та розробники можуть використовувати для швидкого та простого створення (та налаштування) програм. Ця модель дозволяє розробникам зосередитись на прикладному програмному забезпеченні без

необхідності керувати операційними системами, оновленнями програмного забезпечення та іншими інфраструктурними питаннями. IT-менеджери повинні думати лише про управління самим прикладним програмним забезпеченням [5].

Програмне забезпечення як послуга (SaaS) є найбільш поширеним типом хмарних послуг і надає такі програми, як електронна пошта, текстові редактори, програмне забезпечення для спільної роботи, програмне забезпечення для дизайну та ряд інших програм. Програми SaaS зазвичай доступні безпосередньо через веб-браузер, що усуває необхідність встановлення програм на окремих робочих станціях. Операційні системи, програми, дані, сервери, сховища та інше – все це управляється постачальником, тому IT-менеджеру залишається лише турбуватися про використання програмного забезпечення та надання доступу співробітникам.

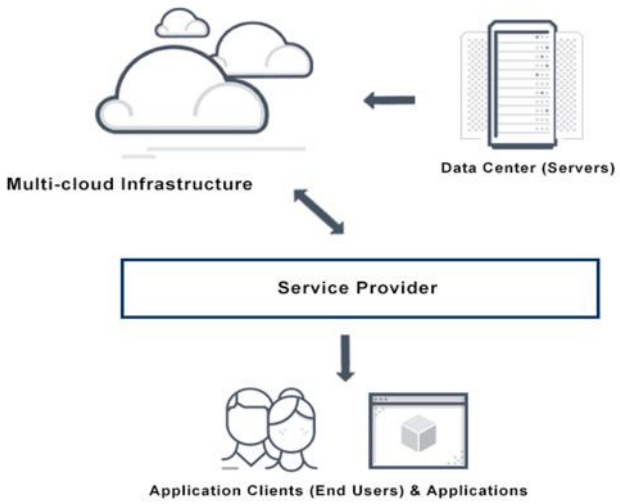

Рисунок 1.3 – Схема розповсюдження IAAS інфраструктури

Основні відмінності між моделями IaaS, PaaS та SaaS полягають у контролі та вартості.

При використанні SaaS у компаній менше контролю, оскільки постачальник управляє додатками, даними, операційними системами, системами зберігання даних, мережами тощо. але вартість відносно нижча. SaaS ідеально

підходить для невеликих компаній, які бажають використовувати хмарні обчислення для зниження витрат, а також для компаній, що працюють у галузях із мінливим попитом.

PaaS пропонує компаніям контроль за їх додатками та даними, в той час як постачальник управляє операційними системами, проміжним програмним забезпеченням, часом виконання і т.д., а вартість є помірною. PaaS ідеально підходить для компаній, які хочуть створювати та контролювати свої власні програми без проблем та складнощів, пов’язаних із створенням мереж, запуском серверів та кодуванням з нуля.

Модель IaaS пропонує компаніям більш повний контроль над додатками та інфраструктурою без необхідності інвестувати у фізичні сервери, мережі та сховища (про які дбає постачальник). Вартість вища, ніж у інших моделях хмарних обчислень, через більшу складність послуг.

Послуги приватної хмари надаються із центру обробки даних компанії внутрішнім користувачам. При використанні приватного хмара організація створює та обслуговує власну базову хмарну інфраструктуру. Ця модель пропонує універсальність та зручність хмар, зберігаючи при цьому управління, контроль та безпеку, характерні для локальних центрів обробки даних. Внутрішнім користувачам можуть виставлятися або не виставлятись рахунки за послуги через систему IT-платежів.

У моделі публічного хмари постачальник хмарних послуг надає хмарні послуги через Інтернет. Послуги публічного хмари продаються на запит, як правило, щохвилини або почасово, хоча для багатьох послуг доступні довгострокові зобов’язання. Клієнти платитимуть лише за цикли процесора, сховище чи пропускну здатність, що вони споживають.

Гібридна хмара – це поєднання загальнодоступних хмарних сервісів та локальної приватної хмари з оркеструванням та автоматизацією між ними. Компанії можуть запускати критично важливі робочі навантаження або важливі додатки в приватному хмарі та використовувати загальнодоступне хмару для обробки сплесків робочого навантаження або стрибків попиту. Метою

гібридного хмари є створення єдиної, автоматизованої, масштабованої середовища, яка використовує всі переваги інфраструктури загальнодоступного хмари, зберігаючи при цьому контроль над критично важливими даними.

Крім того, організації все частіше використовують мультихмарну модель, або використання кількох постачальників IaaS. Це дозволяє програмам мигрувати між різними хмарними провайдерами або навіть працювати одночасно з двома або більше хмарними провайдерами.

Організації переходять на багатохмарну модель з різних причин. Наприклад, вони можуть зробити це, щоб мінімізувати ризик збою в роботі хмарних сервісів або скористатися перевагами конкурентоспроможних цін у певного провайдера. Використання багатохмарних технологій та розробка програм можуть бути складним завданням через відмінності між послугами хмарних провайдерів та програмними інтерфейсами програм (API).

Однак розгортання кількох хмар має стати простішим, оскільки послуги та API постачальників зближуються та стають більш стандартизованими завдяки відраслевим ініціативам, таким як Відкритий інтерфейс хмарних обчислень.

Хмара спільноти, яка спільно використовується кількома організаціями, підтримує певну спільноту, яка має спільні проблеми (наприклад, однакові завдання, політика, вимоги до безпеки та дотримання нормативних вимог). Хмара спільноти управляється або цими організаціями, або стороннім постачальником і може бути як на території, так і за її межами.

Через конкуренцію постачальники повинні постійно розширювати спектр своїх послуг, щоб відрізнятись від інших. Це призвело до того, що публічні постачальники IaaS пропонують набагато більше, ніж звичайні примірники обчислювальних систем та систем зберігання даних.

Наприклад, безсерверні або керовані подіями обчислення - це хмарні послуги, які виконують певні функції, такі як обробка зображень та оновлення баз даних. Традиційні хмарні розгортання вимагають від користувачів створення обчислювального копію та завантаження коду до цього копія. Потім користувач вирішує, як довго працювати – і платити за цей примірник.

При безсерверних обчисленнях розробники просто створюють код, а постачальник хмарних послуг завантажує та виконує цей код у відповідь на реальні події, тому користувачам не потрібно турбуватися про сервер або примірник розгортання хмар. Користувачі платитимуть лише кількість транзакцій, виконуваних функцією.

Публічні хмарні обчислення також добре підходять для обробки великих даних, які вимагають величезних обчислювальних ресурсів протягом коротких періодів години. Хмарні провайдери відповіли на це обслуговуванням обробки великих даних [6].

Ще одна група нових хмарних технологій та послуг пов’язана зі штучним інтелектом (ШІ) та машинним навчанням. Ці технології надають цілий ряд хмарних, готових до використання сервісів ІІ та машинного навчання для потреб клієнтів. Amazon Machine Learning, Amazon Lex, Amazon Polly, Cloud Machine Learning Engine та Google Cloud Speech API – ось приклади таких сервісів.

Багато великих компаній використовують поєднання традиційного обладнання та хмарних обчислень, але все частіше малі та середні підприємства також переходять на хмарні послуги. В залежності від потреб споживач може вибрати один або всі три типи хмарних послуг.

Відсутність стандартів для конфігурації та управління хмарними сервісами, як і раніше, є серйозною перешкодою на шляху компаній, які розглядають можливість впровадження IaaS, оскільки корпоративним користувачам може знадобитися управління гібридною хмарною средою, що включає як приватне, так і загальнодоступне хмару. Оскільки кожен постачальник послуг IaaS надає аналогічні послуги хмарних обчислень, можна розробити загальну модель кожної категорії послуг.

У моделі IaaS є різні типи:

- тип ресурсу, наприклад, обчислення, сховище;
- тип відображення, такий як зображення та знімок;
- інші типи, такі як облікові дані та брандмауер.

У моделі обслуговування пропонується низка служб, включаючи службу

управління ресурсами, службу віртуальних машин, службу клонування та службу безпеки.

Для приватного хмари корпоративним користувачам зазвичай потрібно керувати кластером віртуальних машин. Вони також керують на різних рівнях, від усього кластера до кожної конкретної віртуальної машини. Для підтримки корпоративного використання кластерів загальна модель IaaS була розширена і тепер включає моделювання кластера віртуальних машин.

Віртуальна машина відноситься до типу ресурсів і є сукупністю декількох обчислень, сховищ і мереж. Набір віртуальних машин формує групу серверів, а кластер складається з кількох груп серверів. Служба кластерів керує життєвим циклом кластера. Користувач може створювати, запускати, зупиняти та завершувати роботу всього кластера чи певної групи серверів [7].

Також користувачі стикаються із проблемою вибору вендора IAAS інфраструктури. Проблема ускладнюється тим, що кінцевий користувач зазвичай не має прямого доступу до конфігурації серверного обладнання, а також має обмежені можливості зміни налаштувань.

Найчастіше інфраструктура хмари передбачає використання більш ніж єдиного фізичного чи віртуального сервера. Дані, що зберігаються у хмарі, розподіляються між цими серверами автоматично та прозрачно користувача, тобто. реалізується стратегія балансування завантаження окремих серверів.

Питання відмови стійкості хмарної інфраструктури слід розглядати комплексно, з обов’язковим врахуванням пропускної спроможності комп’ютерної мережі.

Для деяких корпоративних клієнтів це може виявитися складним питанням, оскільки жодна компанія не має однакових потреб. Однак можна скласти список узагальнених факторів, на які слід звернути увагу під час вибору постачальника IaaS. Деякі з них будуть більш важливими для конкретного бізнесу, ніж інші, але кожен з них є важливим базовим фактором, який повинен очікуватися від якісного постачальника IaaS:

- надійна інфраструктура: повністю керований постачальник хмарних

послуг IaaS повинен мати надежну інфраструктуру, здатну забезпечити безпечний доступ до надежних програм, програмного та апаратного забезпечення в центрі обробки даних, який вправно керується та обслуговується;

- передовий досвід ідеально шукати у провайдера, який базується на стандартах ITIL v3. Наприклад, постачальник повинен здійснювати моніторинг та управління мережею цілодобово, використовуючи автоматизовані процеси та складні програмні інструменти;

- гнучкі потужності: Для тих компаній, які стикаються із сезонними коліваннями попиту, постачальник IaaS повинен пропонувати гнучкі потужності, здатні розширюватися та скорочуватися відповідно до цих потреб, з гибкими цінами;

- безпека: Незалежно від того, яку модель IaaS-обчислень ви шукаєте - публічна або приватна хмара, безпека має бути на першому місці. Це особливо важливо для підприємств, які працюють у галузях, де безпека даних регулюється федеральними законами;

- функціональність: це стосується самої хмарної інфраструктури та здатності провайдера запропонувати базову підтримку обчислень та зберігання даних. Це стосується публічних, приватних та гібридних хмарних рішень;

- ынтеграція: Вважно вивчіть кроки та процеси, за допомогою яких постачальник хмарних послуг впроваджує свої послуги у існуючі робочі процеси. Вони повинні відповідати існуючим робочим процесам та майбутньому напрямку розвитку компанії;

- ынтероперабельність: Цей фактор впливає на загальну продуктивність, гнучкість та масштабованість. Важливо, щоб обмін даними між різними системами відбувався гладко, щоб не було збоїв у роботі та щоб користувачі не плуталися;

- зручність використання: Нарешті, перехід на хмарну обчислювальну платформу постачальника IaaS не повинен здійснюватися за рахунок зручності використання. Співробітники повинні легко орієнтуватися в різних системах та програмах.

1.2 Засоби захисту даних

В даний час особливу популярність набули гібридні інфраструктури, які дозволяють настроїти необхідні послуги під конкретні завдання кінцевого користувача. Головною складністю цього підходу є необхідність кваліфікованих кадрів для налаштування та підтримки системи [8].

Для багатьох представників бізнесу одним з головних критеріїв роботи системи буде відмовностійкість та захищеність системи.

Під захищеністю можна становити такі загрози як:

- втрата чи витік;
- крадіжка облікового запису чи служби;
- небезпечний інтерфейс;
- відмова у обслуговуванні;
- шкідливий інсайдер;
- витік даних;
- зловживання хмарними послугами;
- недостатня належна;
- міграція небезпечних віртуальних машин.

Для вирішення цієї проблеми сучасні інфраструктури використовують мережеві екрани (брандмауери).

Брандмауер – це пристрій безпеки мережі, який відстежує вхідний та вихідний мережний трафік і дозволяє або блокує пакети даних на основі набору правил безпеки. Його мета – встановити бар’єр між вашою внутрішньою мережею та вхідним трафіком із зовнішніх джерел (наприклад, Інтернет), щоб заблокувати шкідливий трафік, такий як віруси та хакери (рисунок 1.4).

Брандмауери ретельно аналізують вхідний трафік на основі заздалегідь встановлених правил і фільтрують трафік, що надходить із незахищених або підозрілих джерел, щоб запобігти атакам. Брандмауєри охороняють трафік у точці входу комп’ютера, яка називається портами, де відбувається обмін інформацією із зовнішніми пристроями. Наприклад, «Адреса джерела 172.18.1.1

може досягати пункту призначення 172.18.2.1 через порт 22».

Розглядати IP-адреси можна як будинки, а номери портів як кімнати всередині будинку. Лише довіреним людям (адреси джерела) взагалі дозволено входити в будинок (адресу призначення) – потім він додатково фільтрується, щоб людям у будинку дозволів доступ лише до певних кімнат (портів призначення), залежно від того, є вони власником, дитина чи гість. Власник допускається до будь-якого приміщення (будь-якого порту), а діти та гості допускаються до певного набору номерів (конкретні порти).

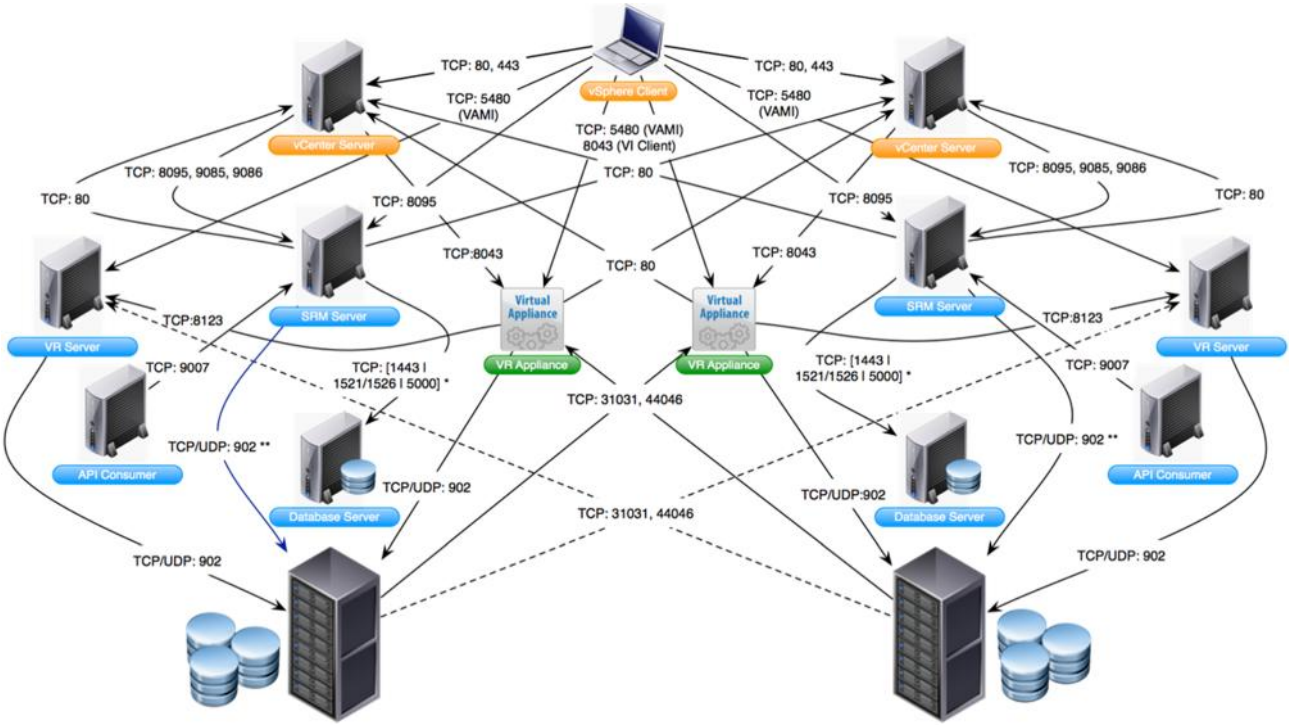

Рисунок 1.4 – Схема інфраструктури з брандмауерами

Брандмауери можуть бути програмними або апаратними, хоча краще мати обидва. Програмний брандмауер – це програма, встановлена на кожному комп'ютері та регулює трафік через номери портів та програми, тоді як фізичний брандмауер – це частина обладнання, встановленого між вашою мережею та шлюзом [9].

Брандмауери з фільтрацією пакетів, найпоширеніший тип брандмауера,

перевіряють пакети та забороняють їх проходження, якщо вони не відповідають установленому набору правил безпеки. Цей тип брандмауера перевіряє IP-адреси джерела та призначення пакету. Якщо пакети відповідають правилам «дозволеного» брандмауера, він довіряє вхід у мережу.

Брандмауери з фільтрацією пакетів поділяються на дві категорії: із збереженням стану та без збереження стану. Брандмауери без стану досліджують пакети незалежно один від одного і не мають контексту, що робить їх легкою мішенню для хакерів. На відміну від цього, брандмауери з визначенням стану запоминають інформацію про раніше передані пакети і вважаються набагато безпечнішими.

Хоча брандмауери з фільтрацією пакетів можуть бути ефективними, вони в кінцевому підсумку забезпечують дуже простий захист і можуть бути дуже обмеженими – наприклад, вони не можуть визначити, чи вплине вміст запиту, який надсилається, негативно на програму, до якої він звертається. Якщо зловмисний запит, дозволений з довіреної адреси джерела, приведе, скажімо, до видалення бази даних, брандмауер не зможе це знати. Брандмауєри наступного покоління та брандмауєри проксі-сервера мають більші можливості для виявлення таких загроз.

Брандмауєри нового покоління (NGFW) поєднують традиційну технологію брандмауера з додатковими функціями, такими як перевірка зашифрованого трафіку, система запобігання вторгненням, антивірус тощо. Зокрема, він включає глибоке перевірення пакетів (DPI). Хоча базові брандмауери переглядають лише заголовки пакетів, глибока перевірка пакетів перевіряє дані у самому пакеті, що дозволяє користувачам ефективніше ідентифікувати, класифікувати або зупиняти пакети зі вредними даними. Дізнайтесь про Forcepoint NGFW тут.

Брандмауери прокси фільтрують мережевий трафік на рівні програми. На відміну від базових брандмауерів, проксі є посредником між двома кінцевими системами. Клієнт повинен надіслати запит до брандмауера, де він потім оцінюється на відповідність набору правил безпеки, а потім дозволяється або

блокується. Зокрема, брандмауери прокси відстежують трафік для протоколів рівня 7, таких як HTTP і FTP, і використовують як перевірку стану, так і глибоку перевірку пакетів для виявлення шкідливого трафіку.

Брандмауери трансляції мережевих адрес (NAT) дозволяють кільком пристроям з незалежними мережевими адресами підключатися до Інтернету за допомогою однієї IP-адреси, зберігаючи приховані окремі IP-адреси. В результаті зловмисники, які сканують мережу на наявність IP-адреси, не можуть отримати конкретні деталі, забезпечуючи більший захист від атак. Брандмауери NAT схожі на брандмауери прокси-сервера тим, що вони діють як посредник між групою комп'ютерів і зовнішнім трафіком.

Брандмауери з багаторівневою перевіркою стану (SMLI) фільтрують пакети на мережевому, транспортному та прикладному рівнях, порівнюючи їх із відомими довіреними пакетами. Як і брандмауери NGFW, SMLI також перевіряє весь пакет і дозволяє їм пройти, лише якщо вони передають кожен рівень окремо. Ці брандмауери перевіряють пакети, щоб визначити стан зв'язку (тому назву), щоб переконатися, що всі ініційовані комунікації відбуваються лише з надійними джерелами.

Брандмауер захищає від різноманітних видів активності у сучасній Інтернет мережі.

Фішинг. Кіберзлочинці розповсюджують посилання на фішингові сайти, які як дві краплі води схожі на ваш онлайн-банкінг, соціальну мережу або ресурс відомого бренду. На таких сайтах пропонується залишити особисті дані – з ними злочинці можуть вивести всі гроші з вашого банківського рахунку або шантажувати особистими фото. Брандмауер припинить підключення до таких ресурсів.

Доступ через бекдор. Так називають уразливості, які іноді залишають – спеціально чи ні – в операційних системах та іншому ПЗ. Це дозволяє кіберзлочинцям або спецслужбам відправляти дані на підключення до Мережі пристрій та приймати трафік з нього, наприклад персональну інформацію, паролі тощо. Брандмауер здатний убезпечити від таких витоків.

Зламування з використанням віддаленого робочого столу. Атаки такого формату дозволяють отримати доступ до комп’ютера по мережі та керувати ним. Брандмауер помітити підозрілий трафік та заборонити його передачу.

Переадресація пакетів. Іноді шахраї змінюють маршрут руху трафіку в Мережі, щоб обдурити систему – змусити повірити, що дані приходять з довіреного джерела. Брандмауер відстежити таке та перекриє канал трафіку.

DDoS-атаки. Якщо брандмауер виявить занадто багато пакетів, які надходять від порівняно маленького списку IP-адрес, він спробує відфільтрувати їх. До речі, якщо комп’ютер спробують використовувати для DDoS-атак, міжмережевий екран також заблокує вихідний трафік.

Щоб міжмережевий екран міг аналізувати дані, він має коректно розпізнавати їх. Зазвичай, брандмауери працюють на кількох рівнях моделі OSI (The Open Systems Interconnection model): мережевому, канальному, транспортному, прикладному та інших. На кожному їх застосовуються свої фільтри. І якщо, наприклад, на канальному (вищому) трафік відповідає правилам, а на прикладному (нижчому) він зашифрований, то брандмауер пропустить такі дані. І це потенційно може призвести до проблем у системі [10].

Не справляються фаєрволи і з тунельованим трафіком - тим, що передається через VPN та інші подібні програми. У цьому випадку між двома точками мережі створюється захищений тунель, одні мережеві протоколи упаковуються в інші (зазвичай нижчий рівень). Брандмауер не може тлумачити такі пакети. І якщо він працює за принципом "все, що не заборонено, дозволено", то пропускає їх.

Зрештою, якщо на ваш комп’ютер зміг проникнути вирус, то брандмауер нічого не зробить із руйнуваннями, які той може викликати. Наприклад, якщо вредне програмне забезпечення зашифрує або видаляє файли або збереже ваші особисті дані, щоб передати їх через зашифрований месенджер або інший захищений канал, фаєрвол навряд чи попередить це.

Зрозуміло, брандмауери вдосконалюються: у них використовують інтелектуальні алгоритми та евристики, які дозволяють виявити проблеми навіть

без заздалегідь підготовлених правил та шаблонів. До того ж у зв’язці з антивірусним програмним забезпеченням фаєрвол блокує значну частину атак. З іншого боку, кіберзлочинці теж не сидять склавши руки і винаходять нові способи обходу захисту.

Також брандмауери поділяють на апаратні та програмні. Переваги та недоліки кожного типу дають можливість вибрати потрібні виходячи з вимог системи. У апаратних міжмережевих екранів спостерігаються такі переваги:

- відносна простота розгортання та використання. Користувач підключив, увімкнув, поставив параметри через веб-інтерфейс і забув про його існування. Втім, сучасні програмні міжмережеві екрани підтримують розгортання через ActiveDirectory, що вимагає багато часу. Але, по-перше, не всі брандмауери підтримують ActiveDirectory, і, по-друге, не завжди на підприємствах використовується Windows як операційна система;
- розміри та енергоспоживання. Як правило, апаратні брандмауери мають скромніші розміри та менше енергоспоживання. Не завжди, щоправда, енергоспоживання відіграє роль, а розміри важливі. Одна справа – невелика компактна коробочка, інша – величезний системний блок;
- продуктивність. Зазвичай продуктивність апаратного рішення вище. Хоча б тому, що апаратний міжмережевий екран займається лише своєю безпосередньою функцією – фільтруванням пакетів. На ньому не запущені будь-які сторонні процеси та служби, як це часто буває у випадку з програмними брандмауерами. Ось уявіть, що ви організували програмний шлюз (з функціями міжмережевого екрану та NAT) на базі сервера з Windows Server. Навряд чи ви виділятимете цілий сервер тільки під брандмауер і NAT. Це нераціонально. Швидше за все, на ньому будуть запущені й інші служби – AD, DNS і т.д. Вже мовчу про СКБД та поштові служби;
- надійність. Апаратні рішення більш відмовостійкі (саме тому, що на них рідко виконуються сторонні служби). Але ніщо не заважає виділити окремий системний блок (нехай навіть не найсучасніший), встановити на нього FreeBSD (одна з найнадійніших у світі операційних систем) та налаштувати правила

брандмауера. Думаю, надійність такого рішення буде не нижчою, ніж у випадку з апаратним файрволом. Але таке завдання потребує підвищеної кваліфікації адміністратора, саме тому раніше було зазначено, що апаратні рішення простіші у використанні.

До переваг програмних рішень належать:

- вартість. Ціна програмного міжмережевого екрану зазвичай нижча за залізку. За вартість середнього апаратного рішення можна захистити всю мережу програмним брандмауером;

- можливість захисту мережі зсередини. Не завжди погрози виходять ззовні. Усередині локальної мережі є безліч загроз. Атаки можуть виходити із внутрішніх комп'ютерів. Ініціювати атаку може будь-який користувач LAN, наприклад, незадоволений компанією. Як зазначалося, можна, звісно, використовувати окремий апаратний маршрутизатор для захисту кожного окремого вузла, але практично таких рішень не зустрічалися. Аж надто вони нераціональні;

- можливість розмежування сегментів локальної мережі без виділення підмереж. Найчастіше до локальної мережі підключаються комп'ютери різних відділів, наприклад, бухгалтерії, фінансового відділу, IT-отдела тощо. Не завжди ці комп'ютери мають взаємодіяти між собою. Для виконання розмежування ІСПДН э декілька можливих рішеннь. Перше рішення полягає у створенні декількох підмереж (наприклад, 192.168.1.0, 192.168.2.0 і т.д.) та налаштуванні належним чином маршрутизації між цими підмережами. Не можна сказати, що рішення дуже складне, але все ж таки складніше, ніж використання програмного файрвола. Та й не завжди можна виділити підмережі з тих чи інших причин. Друге рішення полягає у використанні міжмережевого екрану, призначеного саме для захисту ІСПДн (не у всіх програмних міжмережевих екранах легко розмежувати ІСПДн). У цьому випадку навіть у найбільшій мережі ви виконаєте розмежування ІСПДн за лічені хвилини, і не доведеться витрачати час на налаштування маршрутизації;

- можливість розгортання на серверах. Якщо є достатній комп'ютерний

парк. Достатньо на одному із серверів розгорнути міжмережевий екран та налаштувати NAT та маршрутизацію. Зазвичай обидві операції виконуються за допомогою графічного інтерфейсу міжмережевого екрану і реалізуються за допомогою декількох клацань мишею в потрібних місцях;

- розширений функціонал. Як правило, функціонал програмних міжмережевих екранів ширше, ніж у апаратних побратимів. Так, деякі міжмережеві екрани надають функції балансування навантаження, IDS/IPS тощо корисні речі, дозволяють підвищити загальну безпеку системи обробки даних. Так, такі функції є не у всіх програмних брандмауерів, але ніщо і ніхто не заважає вам вибрати міжмережевий екран, що відповідає вашим потребам. Звичайно, такі функції мають і деякі апаратні комплекси. Наприклад, StoneGate IPS – надає функціонал системи запобігання вторгненням, але вартість таких рішень не завжди сподобається керівництву підприємства. Також є і апаратні балансувальники навантаження, але вони коштують ще дорожче, ніж апаратні IPS.

Недоліки випливають із переваг. Переваги одного виду брандмауерів, зазвичай, є недоліками іншого виду. Наприклад, до недоліків апаратних рішень можна віднести вартість і неможливість захисту локальної мережі зсередини, а до недоліків програмних – складність розгортання та використання (хоча, як було зазначено, відносно).

Як правило, у всіх апаратних міжмережевих екранів є кнопка скидання, натиснувши яку можна повернути стандартні параметри. Для натискання цієї кнопки не потрібно мати якусь особливу кваліфікацію. А ось, щоб змінити параметри програмного міжмережевого екрану, потрібно як мінімум отримати права адміністратора. Натиснувши одну кнопку, незадоволений співробітник може порушити безпеку всього підприємства (або залишити підприємство без доступу до Інтернету, що краще). Тому при використанні апаратних рішень необхідно більш відповідально підійти до питань фізичної безпеки самих пристроїв [11].

Також для забезпечення цілісності системи та її внутрішніх даних

розглядається політиками конфіденційності та захисту даних. Під стабільністю роботи передбачається апаратна частина інфраструктури, яка може вийти з ладу і кінцевий користувач не може виправити цю проблему власноруч.

1.3 Відновлення роботи вузлів системи

Оскільки відмовостійкість хмарної інфраструктури ґрунтується на підвищенні індивідуальної надійності кожного окремого сервера. Загроза загальних уразливостей існує у всіх моделях, оскільки базові компоненти, що розгортають інфраструктуру, платформи та програми, не забезпечують надійної ізоляції між моделями хмарних обчислень. Оскільки кожен з процесів виконується в рамках ізольованої середовища, потрібно розробити модуль, який буде контролювати стан всіх компонентів інфраструктури і в разі потреби, такий як вихід вузла з ладу, переадресовувати завдання на інший вільний узел, а також зробити спробу перезапуску вузла, що вийшов з ладу і в разі неуспішної операції перезапуску відправлення звіту персоналу.

1.3 Постановка задачі

Таким чином, метоє кваліфікаційної роботи є розробка методу створення відмовостійких багатошарових віртуальних інфраструктур в хмарному середовищі для забезпечення стабільної роботи системи, який би виправляв основні недоліки сучасних підходів щодо організації віртуальних інфраструктур та тестування стабільності їхньої роботи.

2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

2.1 Адаптивний генетичний алгоритм з недомінованим сортуванням

Кожен глобальний вендор IAAS інфраструктур так чи інакше піклується про стійкість до відмови системи, але кінцевий користувач не має гарантій, що обрані рішення виявлятимуться вдалими за різного роду збоїв (рисунок 2.1).

Тому в даний час науковці, які займаються роботою з віртуальними мережами, все частіше досліджують проблему створення методів відмовостійкості у різноманітних віртуальних системах. Їх рішенняпублікуютьсяна наукових конференціях та за їх технічними рішеннями та аналізу результатів можна робити висновки [12].

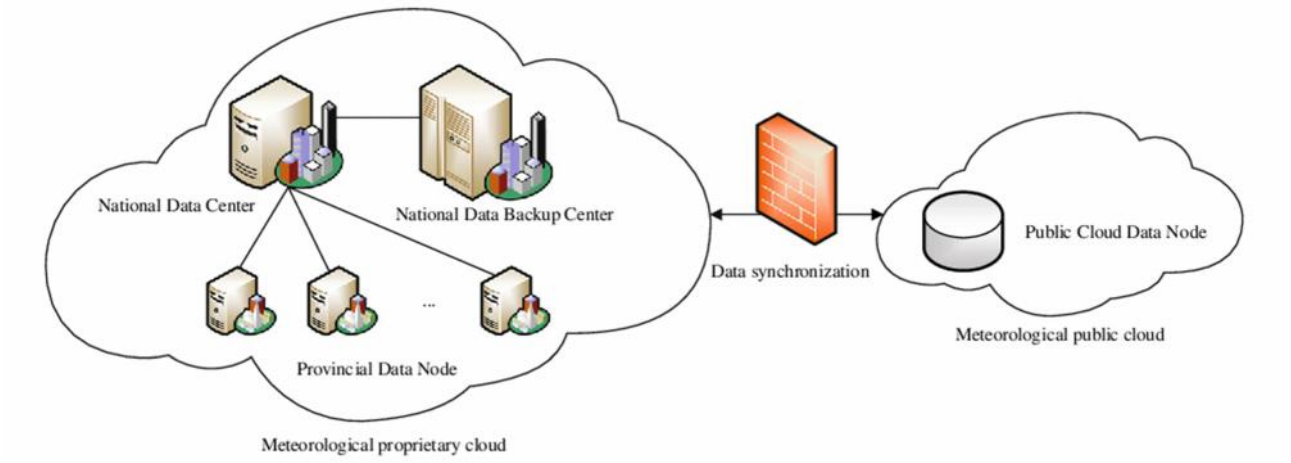

Рисунок 2.1 – Адаптивний генетичний алгоритм з недомінованим сортуванням

Одним із таких рішень розроблено для бездротових сенсорних мереж, у якому пропонується структура для оптимізації стійкості до відмов (FT) при віртуалізації у WSNs, орієнтована на гетерогенні мережі для сервіс-орієнтованих додатків IoT. Оскільки відмова каналу зв’язку у WSNs впливає безліч віртуальних мереж, що працюють із сервісами IoT. Сформульовано завдання

оптимізації, що розглядає відмовостійкість та затримку зв’язку як дві конфліктуючі цілі. Для вирішення задачі оптимізації було розроблено адаптивний генетичний алгоритм із недомінованим сортуванням (A-NSGA). Основні компоненти A-NSGA включають обчислення відмовостійкості та затримки, кросинговер та мутацію, а також сортування на основі недомінування. Автор представляє аналітичну та імітаційну порівняльну оцінку ефективності.

Недоліком цього рішення є те, що запропонована структура використовує алгоритм із недомінованої сортівки, що тягне за собою необхідність додаткових технічних потужностей для безперебійного прорахунку [13].

2.2 Програмне рішення AESON

Ще одне рішення є використанням програмного рішення AESON (Activation Engine on Service Overlay Network). AESON – середовище виконання розгортання, яка автоматично виявляє збої вузлів (віртуальних машин) і зрештою призводить до складового сервісу у бажаний стан розгортання, використовуючи інформацію, що описує відносини між компонентами сервісу (рисунок 2.2).

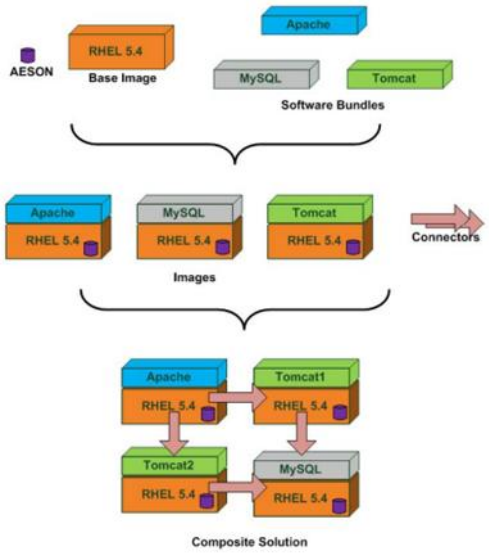

Рисунок 2.2 – Процес створення образів та моделювання композитної послуги

Перевагою цього рішення є автоматизація настроювання інфраструктури, т.к. Типові програми, що розгортаються в хмарах IaaS, потребують складних програмних стеків, що складаються з безлічі розподілених, взаємозалежних компонентів, таких як балансувальники навантаження, веб-сервери, сервери додатків та баз даних. Під час розгортання всі необхідні програмні компоненти та проміжне програмне забезпечення повинні бути налаштовані для забезпечення коректної роботи програми, а вся інфраструктура також має бути налаштована.

Вся інфраструктура також має бути налаштована для підтримки нефункціональних вимог, таких як продуктивність, доступність та безпека.

Розгортання та правильне налаштування таких додатків є нетривіальним завданням, що загрожує помилками. Враховуючи величезні масштаби центрів обробки даних провайдерів IaaS провайдерів, які складаються з великої кількості різноманітних апаратних компонентів, не дивно, що апаратні в таких средах часто відбуваються збої обладнання, в тому числі через людські помилки під час експлуатації дата-центрів, які посилюють цю проблему. Збої в центрах обробки даних IaaS можуть призвести до частичного або повного припинення роботи розміщених додатків, внаслідок чого постраждалі компанії можуть зазнати збитків. На жаль, оскільки хмарний провайдер не має семантичних знань про розміщені додатки та їх топології, неможливо довірити йому відновлення після збоїв, що може спричинити специфічні для застосування дії, включаючи перезапуск деяких його компонентів та зміну конфігурації інших. Отже, частина тяжіння управління все ще лежить на власниках розміщених додатків. Запропоноване рішення може обробляти три типи збоїв: відмова вузла, відмова вузла завісання вузла та відмова компонентів програми [14].

Недоліком цього рішення є його комерційне поширення та деякі додаткові вимоги технічних потужностей. Також до мінусів можна віднести те, що програмне рішення дає можливість лише обробки збоїв, а не аналізувати та попереджати про можливі проблеми.

2.3 Використання багаторівневої стратегії

Ще одним рішенням є використання багаторівневої стратегії, яка проста у реалізації для ефективного аудиту, але вона містить деякі потенційні обмеження (рисунок 2.3). Наприклад, використання менеджера сесій та хмарного менеджера може призвести до єдиної точки відмови. Крім того, існує високий ризик перевантаженості, особливо на диспетчері сесій, оскільки він відповідає за підтримку безпечного сеансу між хмарними сервісами та користувачами шляхом видачі токенів, і існує непередбачувана велика кількість користувачів хмари, які можуть отримати доступ до хмари одночасно.

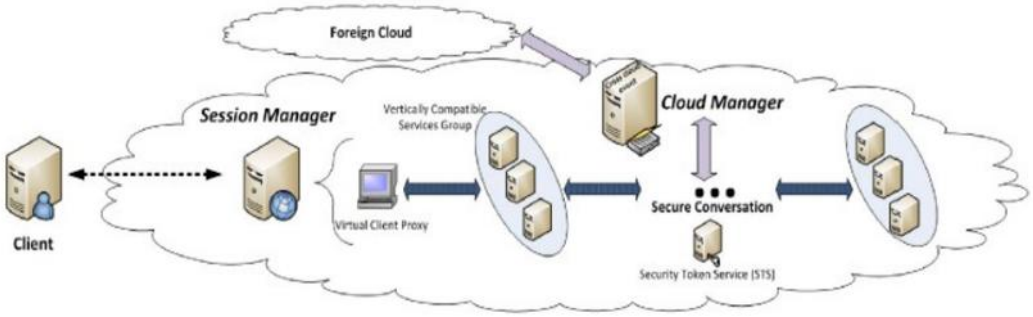

Рисунок 2.3 – Хмарна архітектура та рівні аудиту

Для захисту менеджерів сеансів та менеджерів хмари від будь-яких атак та делегування, перевірки та скасування маркерів для ідентифікації управління ідентифікацією в рамках захищеного сеансу. Менеджер сеансів відповідає за делегування клієнтських запитів хмарним сервісам та реєстрацію історії використання сервісів та заміни. Рівень міжхмарного аудиту (світлі стрілки рису) авторизований за допомогою Cloud Manager для обміну повідомленнями та ресурсами між різними хмарами або зовнішніми сервісами в хмару [15].

Для виявлення уразливостей безпеки аудиторські перевірки можуть бути зроблені з використанням багаторівневої стратегії, що обговорюється. Наприклад, служба маркерів безпеки (STS) Аудиторське випробування

створюється шляхом збирання подій, захоплених STS, оскільки STS відповідає за створення маркера для безпечного сеансу з використанням відомого протоколу обміну повідомленнями. Менеджер сеансу може контролювати очевидні аномалії, наприклад, токен не було скасовано, коли таймінг сервісів може виявити атаку чи проблему продуктивності у журналі аудиту STS Audit Trail у поєднанні з історичними журналами. Журнал аудиту ланцюжка викликів складається з імплантацією частин службового журналу аудиту серед відповідних подій, які зареєстровані в журналі STS Audit Trail. Хмарний журнал аудиту формується за допомогою STS Audit Trail і замінюючи будь-які кросс-доменні точки кросс-хмарними подіями, зареєстрованими менеджером хмари. Питання довіри може виникнути, якщо аномалія, виявлена менеджером сеансів, може бути усунена хмарним менеджером. Хмарний журнал аудиту може вказувати на інші проблеми продуктивності чи безпеки, такі як невиконання службою виконує контракт або викликає потенційні слабкі місця у хмарі та мають бути виправлені чи усунені.

Це рішення найбільше підходить для серверів аутентифікації, т.к. його архітектура дає можливість взаємодіяти з токенами і має високий рівень захисту даних, але недоліком є те, що за наявності великої інфраструктури немає можливості забезпечити стабільну роботу і відновлення в разі падіння окремого вузла.

2.4 Організація покращення відмовостійкості мережевих екранів

Також як аналоги розв’язання цієї проблеми було обрано наукову статтю на тему «Сценарії реалізації технологій вкладеної віртуалізації у задачі покращення відмовостійкості міжмережевого екрану» (рисунок 2.4).

Дане рішення є міжмережевим екраном реалізованим у вигляді набору функціональних програмних модулів, кожен з яких укладений у віртуальний контейнер, які підтримуються гіпервізорами у віртуальній машині. Транскордонний маршрутизатор та балансувальник реалізовані у вигляді

додатків в окремих віртуальних контейнерах, реалізованих за допомогою технології SaaS та розгорнутих на віртуальній машині [16].

Автори наводять результати експериментів, які дають розуміння, що в разі низькорівневої DDos-атак система відновлюється в робочий стан за менше 2 годин після запуску системним адміністратором процесу відновлення. Більше години зайняло розгортання хмарного міжмережевого екрану в умовній хмарі іншого провайдера та повторної прив’язки IP-адреси до домену за допомогою служби динамічного DNS.

Це рішення показало, що цей підхід може скоротити годину відновлення до 7 разів. Хмарний сервер дозволяє робити копії самого себе, що дозволяє підняти всю інфраструктуру до хмар інших провайдерів, скорочуючи годину, коли ресурс взагалі недоступний.

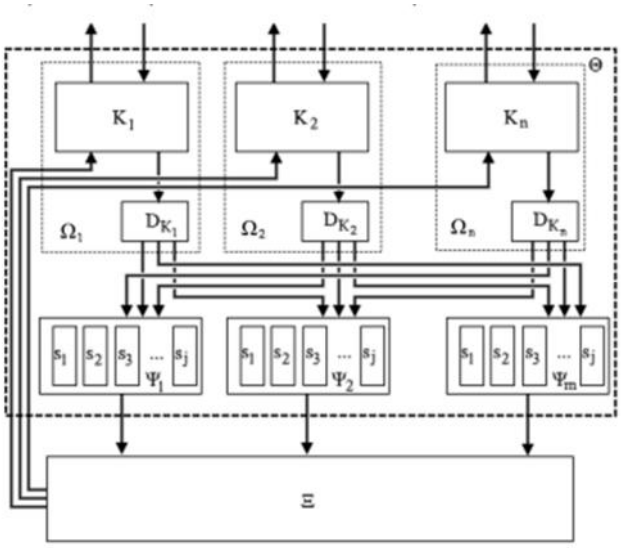

Рисунок 2.4 – Блок-схема хмарного міжмережного екрану

Рішення має також низку недоліків таких як відсутність автоматизації, що потребує цілодобового обслуговування системи. А також неможливість аналізувати трафік із метою захисту від серверних атак.

2.5 Відновлення даних віртуальної системи

Одним із сценаріїв відновлення роботи віртуальної системи також слід розглянути втрату даних.

Більшість організацій, що працюють з пошуком виддалених даних з пристроїв злочинців, використовують метод “чорної скриньки”. Метод полягає в тому, що всередині хостової операційної системи зберігається текстовий лог-файл із метаданими від віртуальної машини. Після видалення віртуальної машини файл залишається на хостовій машині, але файл зберігається в системі навіть після переустановки операційної системи хостової машини. Використовуючи сучасні бібліотеки, існує можливість відновити близько 70% даних.

2.6 Способи розгортання віртуальної інфраструктури

Одним із найпоширеніших інструментів контейнеризації є програмне забезпечення Docker (рисунок 2.5). Він вирішує безліч завдань, пов’язаних із створенням контейнерів, розміщенням у них додатків, керуванням процесами, а також тестуванням ПЗ та його окремих компонентів.

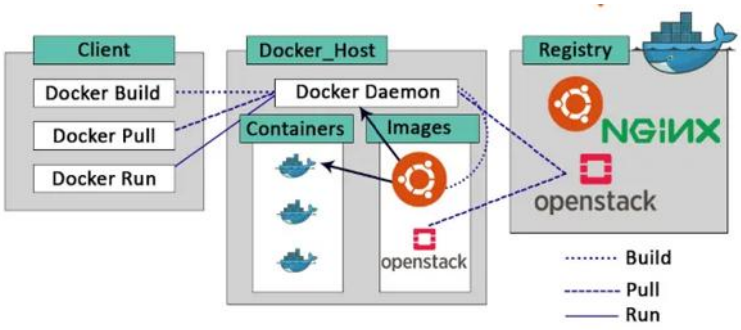

Рисунок 2.5 – Архітектура роботи Docker

Контейнеризація є чудовою альтернативою апаратної віртуалізації. Всі процеси в ній протікають на рівні операційної системи, що дозволяє суттєво

економити ресурси та збільшувати ефективність роботи з додатками [17].

Docker потрібен для більш ефективного використання системи та ресурсів, швидкого розгортання готових програмних продуктів, а також для їх масштабування та перенесення в інші середовища з гарантованим збереженням стабільної роботи.

Основний принцип роботи Docker – контейнеризація програм. Цей тип віртуалізації дозволяє упаковувати програмне забезпечення ізольованими середовищами – контейнерами. Кожен із цих віртуальних блоків містить усі необхідні елементи для роботи програми. Це дає можливість одночасного запуску великої кількості контейнерів на одному хості.

Переваги використання Docker:

- мінімальне споживання ресурсів - контейнери не віртуалізують всю операційну систему (ОС), а використовують ядро хоста та ізолюють програму на рівні процесу. Останній споживає набагато менше ресурсів локального комп'ютера, аніж віртуальна машина;

- швидкісне розгортання – допоміжні компоненти можна не встановлювати, а використовувати вже готові образи (шаблони). Наприклад, немає сенсу постійно встановлювати і налаштовувати Linux Ubuntu. Достатньо 1 раз її встановлювати, створити образ і постійно використовувати лише оновлюючи версію при необхідності;

- зручне приховування процесів – для кожного контейнера можна використовувати різні методи обробки даних, приховуючи процеси фону;

- робота з небезпечним кодом – технологія ізоляції контейнерів дозволяє запускати будь-який код без шкоди для ОС;

- просте масштабування – будь-який проект можна розширити, впровадивши нові контейнери;

- зручний – додаток, що знаходиться всередині контейнера, можна запустити на будь-якому docker-хості;

- оптимізація файлової системи – образ складається з шарів, які дозволяють ефективно використовувати файлову систему.

Аналогом цього рішення є програмне забезпечення Kubernetes. Kubernetes є проектом з відкритим вихідним кодом, який керує кластером контейнерів на операційній системі Linux як однією цілою системою. Програмне забезпечення має змогу керувати процесами роботи Docker-контейнерів на широкому пулі хостів.

Працюючий кластер Kubernetes включає агента, запущеного на нодах (kubelet) і компоненти майстра (APIs, scheduler), поверх рішення з розподіленим сховищем. Наведена схема показує бажаний, зрештою, стан, хоча все ще ведеться робота над деякими речами, наприклад: як зробити так, щоб kubelet (усі компоненти, насправді) самостійно запускався в контейнері, що зробить планувальник на 100% підключається [18].

Docker та Kubernetes – це різні технології. Не зовсім справедливо порівнювати ці два варіанти або ставити питання про те, кому з них слід віддати пріоритет.

Docker – це контейнерний майданчик, а Kubernetes є інструментом оркестрування контейнерів для таких платформ як Docker.

Docker та Kubernetes найчастіше використовують разом. Docker використовується для ізоляції додатків у контейнерах, а Kubernetes як планувальник для розгортання та масштабування.

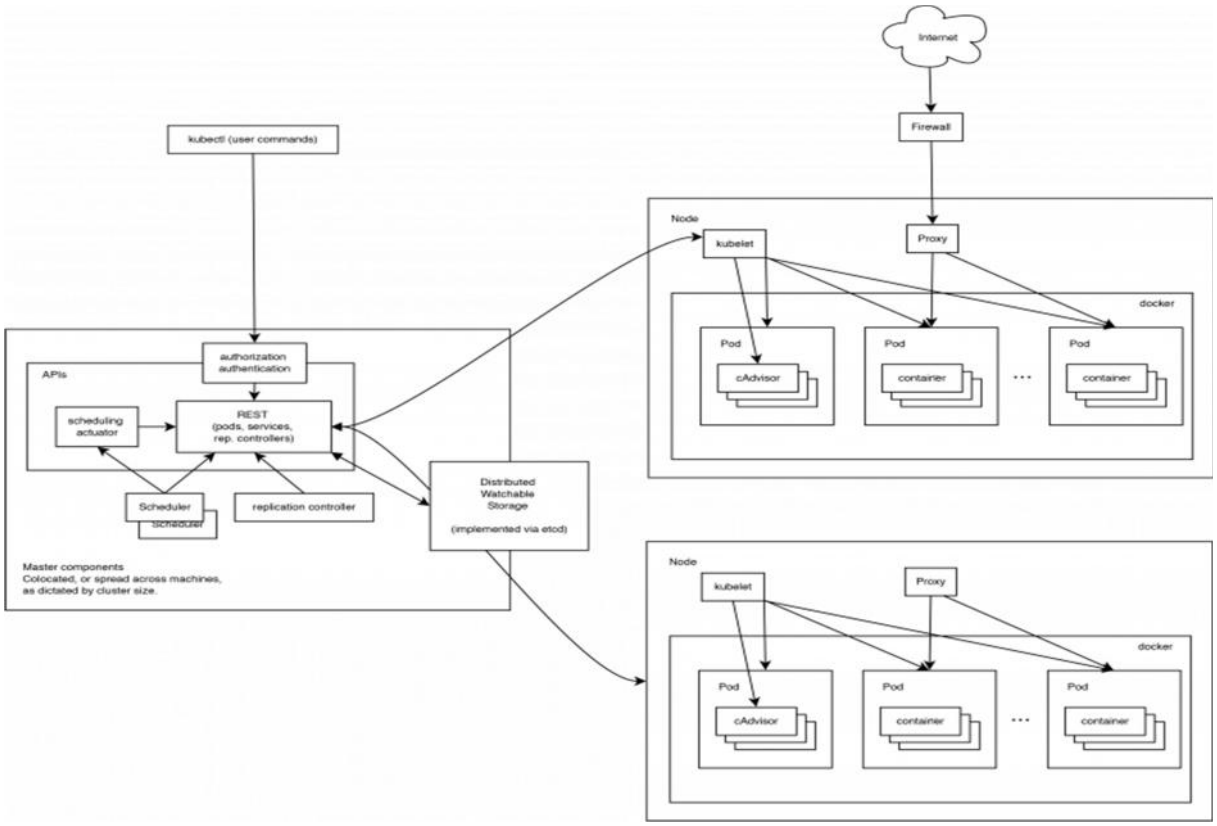

Рисунок 2.6 – Архітектура Kubernetes

Подібності Docker та Kubernetes:

- Архітектура, що базується на мікросервісах.
- В основному написані на мові програмування Go, так що поставляються у вигляді легких бінарників.
- Обидва використовують файли в форматі для людини YAML.

Якщо у вашій роботі використовується мікросервісна архітектура – беріть Docker та призначайте контейнер для кожного мікросервісу. Kubernetes - це технологія, що швидко розвивається, але вона потрібна тільки якщо у вас великий штат розробників, яким потрібна серйозна продакшн-середовище. Для маленького проекту та трьох-чотирьох розробників накладні витрати можуть «з'їсти» всю користь.

Docker потрібно використовувати якщо програма підходить для роботи в контейнері; програма не вимагає графічного інтерфейсу; додаток має бути розгорнуто послідовно.

Kubernetes варто використовувати за умов, що організація не прив’язана до одного хмарного провайдера, то використання цього рішення є найрозумнішим. Причина в тому, що він працює однаково на всіх системах. Ось чому його називають вендор-незалежним [19].

В епоху хмарних сервісів, опенсорсного програмного забезпечення, сервісів/мікросервісів та контейнеризації компаніям вдасться залучити талановитих DevOps-фахівців для розвитку інфраструктури. Контейнери допоможуть зосередитись на бізнес-логіці, не відволікаючись на те, що знаходиться під капотом.

2.7 Висновки аналізу існуючих рішень.

Провівши аналіз існуючих прикладних рішень проблеми організації відмовостійкої архітектури систем віртуалізації, можна виділити, що програмні рішення, що забезпечують встановлення, підтримку роботи та захист від збоїв для віртуальних систем, поширюються на комерційній основі, а також є “чорною скринькою”, що позбавляє системи гнучкості та можливості зміни параметрів з боку користувача та адміністратора. Також одним із недоліків є відсутність механізму аналізу трафіку та попередження підозрілого трафіку.

Більшість методів мають механізм балансування навантаження у разі падіння окремих вузлів системи, але не мають апаратної можливості відновити роботу цих вузлів без втручання технічного персоналу, що може зазнати великих тимчасових затримок у разі відмови системи у позаробочий час, а також інші причини, такі як карантинні обмеження, які можуть обмежити доступ до апаратної частини та серверного терміналу.

3 МЕТОД ОРГАНІЗАЦІЇ ВІДМОВОСТІЙКОЇ БАГАТОШАРОВОЇ ВІРТУАЛЬНОЇ ІНФРАСТРУКТУРИ

3.1 Гнучкість методу

Більшість існуючих рішень на сьогоднішній день є комерційними та працюють за принципом перенаправлення мережевих запитів, використовуючи безпечні маршрути передачі даних. Головним недоліком цих систем є відсутність гнучкості та неможливість вносити корективи до параметрів, т.к. користувач розміщує на орендованих потужностях "чорний ящик", що забезпечує цілісність системи. Запропоноване рішення є гнучким і дозволяє мігрувати між вендорами хмарних послуг, змінювати програми контролю версій тощо.

3.2 Балансування навантажень

Запропоноване рішення передбачає декомпозицію архітектури таким чином, щоб інфраструктура розміщувалася на двох або більше фізичних або віртуальних серверах. Це дозволить у разі виходу з ладу одного з перенаправити завдання на працюючі сервери [20].

Стратегія балансування навантаження окремих серверів дозволить в автоматичному режимі та прозоро для користувача розподіляти дані між цими серверами.

Серед цілей, для досягнення яких використовується балансування, можна виділити такі:

- чітке розподілення: потрібно оранізувати, щоб обробка кожного запиту задіяла усі системні ресурси і не допустити виникнення ситуацій, коли запит обробляється, а решта чекає своєї черги;
- ефективність: усе обладнання повине працювати на повну потужність та

уникати ситуацій, коли один з серверів очикує наступні задачі:

- скорочення часу обробки запиту;
- скорочення часу відповіді.

У рішенні використовується комбінація двох алгоритмів Weighted Round Robin та Least Connections (рисунок 3.1).

Weighted Round Robin- Удосконалена версія алгоритму Round Robin. Суть удосконалень полягає в наступному: в заленжості от продуктивновсті сервірів їм встановлються коєфіцвєнти. Це допомагає розподіляти навантаження гнучкіше: сервери з великою вагою обробляють більше запитів. Недоліком алгоритму є те, що в ньому зовсім не враховується кількість активних на даний момент підключень [22].

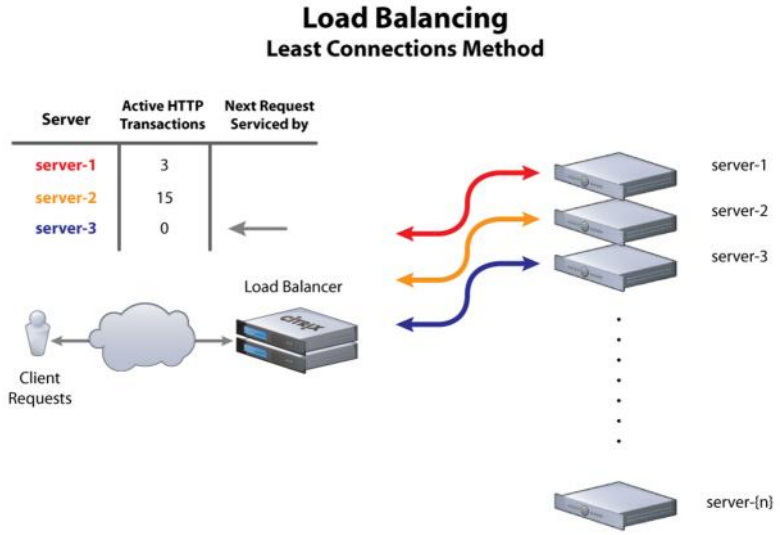

Рисунок 3.1 – Алгоритм Least Connections

Алгоритм Least Connections враховує кількість підключень, що підтримуються серверами в даний час. Кожне питання передається серверу з найменшою кількістю активних підключень.

У разі виходу з ладу одного із серверів балансувальник навантаження перерозподіляє навантаження між активними серверами. Додатково

збільшується тактова частота центральних процесорів, використовуючи технологію AMD Turbo Core та Intel Turbo Boost, завдяки чому система не втрачає в продуктивності. Для забезпечення відмовостійкості та відновлення роботи запропоновано використання методу збереження станів процесу на основі контрольних точок. Даний метод дозволяє відновлювати стан процесів у разі відмови, при цьому процеси обмінюються повідомленнями для контролю стану один одного.

Методи неузгодженого збереження станів дозволяють кожному процесу самостійно створювати контрольні точки. У цьому формується глобальна цілісна структура. Відмовостійкість у цій моделі ґрунтується на надійності кожного обчислювального вузла. Віртуальна машина виконує алгоритм докладання реального часу, після чого запускається модуль перевірки, який відповідає за коректність роботи віртуальної машини.

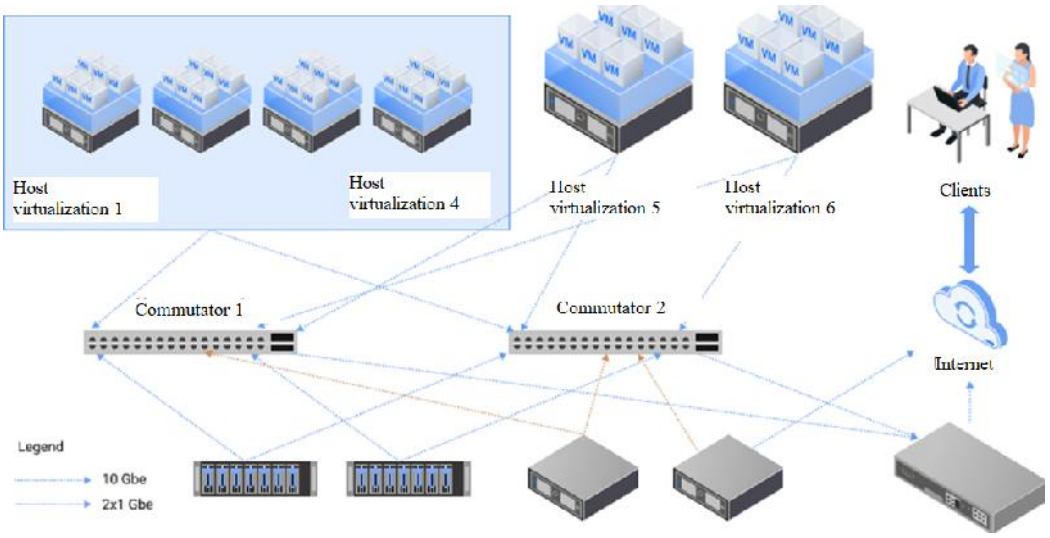

Рисунок 3.2 – Архітектура запропонованого рішення

Модуль перевірки результатів роботи віртуальної машини передає результати виконання завдання модулю перевірки часу, який звіряє з встановленими обмеженнями час виконання.

Модуль оцінки надійності, у свою чергу, розраховує та задає значення

надійності для кожної віртуальної машини. Потім інформація передається у модуль прийняття рішень, який вибирає вихідні дані вузла з найвищою надійністю, та створюється контрольна точка відновлення. Відновлення системи передбачає відкат до контрольної точки несправного процесу. Відновлення повідомлень, відправлених у період між відмовою системи та контрольною точкою, походить з лога, який зберігається в постійній пам’яті і повторно обробляється при передачі користувачеві. Таким чином, відновлений процес повторно генерує копії повідомлень, відправлених до відмови [23].

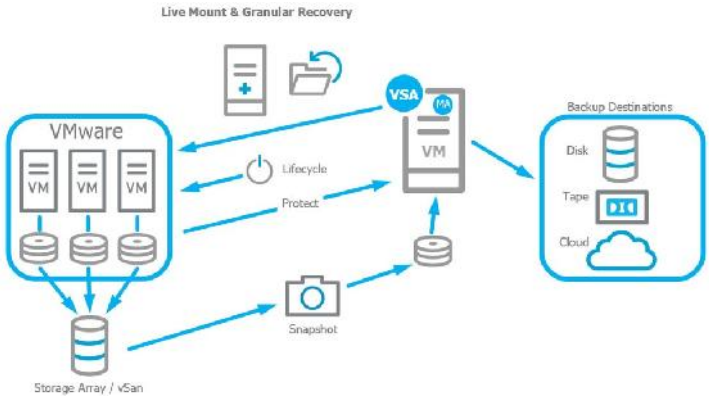

Рисунок 3.3 – Алгоритм відновлення роботи віртуальних систем

Ще одним важливим компонентом архітектури є попередження відмови в роботі системи. У свою чергу, існує кілька сценаріїв, які можуть призвести до відмови системи такі як: вихід з ладу окремих компонентів, проблеми з електропостачанням серверів і різні види атак на серверну частину системи. Оскільки перші два фактори є дуже неконтрольованими та їх прогнозування неможливе, то останній дозволяє створити захист.

Типовим рішенням для цього є використання брандмауера.

3.3 Автоматичне відновлення роботи окремих вузлів

Як система управління репозиторію програмного коду для Git буде використовуватися сайт GitLab.

GitLab пропонує рішення для зберігання коду та спільної розробки масштабних програмних проектів. Репозиторій включає систему контролю версій для розміщення різних ланцюжків розробки та гілок, дозволяючи розробникам перевіряти код і повертатися до стабільної версії софту у разі непередбачених проблем.

GitLab є конкурентом GitHub, у якому серед багатьох інших проектів розміщується розробка ядра Linux Лінуса Торвальдса. Оскільки GitLab розробляється тієї ж основі управління версіями (Git), принцип роботи схожий. GitLab підтримує як публічні, так і необмежену кількість приватних гілок розробки.

Одним із ключових при виборі даного аспекту системи була безперервна інтеграція.

Безперервна інтеграція – це сучасна методологія розробки, за якою код вносяться невеликі зміни з частими комитами. І оскільки більшість сучасних додатків розробляються з використанням різних платформ та інструментів, то виникає потреба в механізмі інтеграції та тестуванні змін, що вносяться.

Мета CI – забезпечити доступний, послідовний та автоматизований спосіб складання, пакування та організації процеса тестування програмних додатків. При цій методології розробники з більшою ймовірністю робитимуть часті комміти, що, у свою чергу, сприятиме покращенню комунікації та підвищенню якості програмного забезпечення за рахунов постійного тестування.

Безперервне постачання починається там, де закінчується безперервна інтеграція. Вона автоматизує розгортання додатків у різні оточення: більшість розробників працюють із продакшн, з середовищами розробки та тестування.

Інструменти CI/CD налаштовують технічні параметри оточення, які конфігуруються перед процесом розгортання системи. Додатково автоматизація дозволяє встановити потрібні комунікації с базами даних та іншима сервісами, які можуть потребувати додатковий час для їх розгортання.

З практики багатьох сучасних компанії підтримка безперервної інтеграції та постачання дозволяють забезпечити безперервного процесу тестування, що

покращують оцінку продукту.

Тестувальники використовують різні види тестування – автоматизовані скрипти на різних мовах програмування для тестування взаємодії користувача, моделюючи його поведінку, які можуть запускатися за допомогою CI/CD-конвеєрів.

Також цей підхід дозволяє після вдалого тестування розгорнути у релізному середовищі.

Багато команд, що використовують CI/CD-конвеєри у хмарах, використовують контейнери, такі як Docker, та системи оркестрації, такі як Kubernetes. Контейнери дозволяють стандартизувати упаковку, постачання та спростити масштабування та знищення оточення з непостійним навантаженням.

Завдяки безперервному розгортанню створено систему відновлення, представляється вона тригером. Цей тригер завдяки опитуванням кожного вузла через час, заданий системним адміністратором, отримує інформацію про їхні стани (рисунок 3.4).

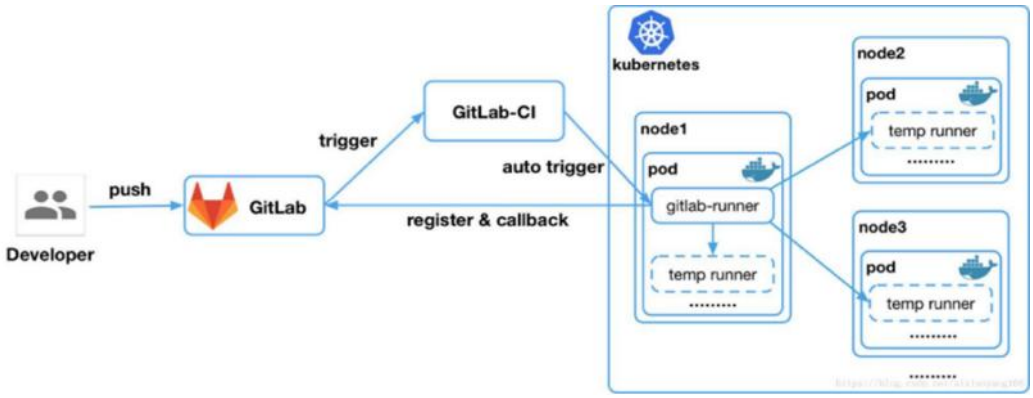

Рисунок 3.4 – Схема алгоритму роботи GitLab CI з тригером

У разі виявлення можливого виходу з ладу вузла, тригер відправляє повторний запит на цей вузол, щоб переконатися, що вузол остаточно вийшов з ладу, а не опитування потрапило в момент перезавантаження чи інших. У разі підтвердження збою в роботі тригер запускає команду, яка зупиняє контейнер і

запускає його заново.

Після перезапуску контейнера проводиться опитування стану вузла, якщо перезавантаження успішне і система знову функціонує, то інформація про збій додається до журналу подій для системного адміністратора з позначкою “Minor”, якщо вузол не відновив працездатність, то системний адміністратор із засобів зв’язку обраними раніше отримує повідомлення про критичну ситуацію та логи помилки. Це дозволяє оперативно втрутитися та усунути неполадки.

Якщо системний адміністратор з тих чи інших причин не втручається в систему після критичної ситуації балансувальник, який використовує активні сервери, а система піднімає тактову частоту процесорів з метою нагнітання потужностей.

Для аналізу приблизного часу відновлення системи потрібно розробити математичну модель, враховуючи, що резервний сервер працює паралельно. Математична модель часу відновлення працездатності вузлів віртуальної інфраструктури враховує, що у всіх випадках при відсутності одного основного вузла та наявності виправного серверу, імовірно, x відбувся перехід на перший резервний елемент.

Імовірність має такий зміст, що з імовірністю x відмова основного елемента виявляється, перемикач виявляється справним і спрацьовує достатньо, причому всі перехідні процеси закінчуються за допустимий час. З вірогідністю $1-x$ при має місце відмова (через затримку в результатах відмов, спрацьовування переміча тощо), випадковий час існування якого (з моменту закінчення допустимого часу до закінчення перехідних процесів) у середньому рівні систем θ (закон розподілу систем перехідного періоду не має значення при $\theta \ll 1/k$). Передбачається, що з ймовірністю x відмова основного елемента виявляється за допустимий час (параметри перемикача не впливають на значення x). З імовірність $1-x$ виникає відмова системи середньої тривалості τ.

Відмова резервного елемента з ймовірністю η виявляється миттєво; з ймовірністю виявлення $1-\eta$ затримується на час, розподілене за експоненційним законом із середнім τ_η. Величини η і τ_η характеризують два

види контролю резерву: безперервного неповного контролю з глибиною η та періодичного повного з випадковим періодом (середнє значення періоду контролю τ_η). Якщо цей контроль неповний з глибиною $\eta_к < 1$, то формули справедливі зі збільшенням τ_η в $1/\eta_к$ розпаду [24].

Розподіл напрацювання між відмовами та час відновлення до елементів експоненційні. Інтенсивність відмов елемента в основному режимі λ, в резерві λ_1 ($\lambda_1 \ll \lambda$) інтенсивність відновлення μ. Передбачається, що невиявлений відмова в резервному елементі, напевно, з'являється при виникненні іншої відмови, який може виявитися системою контролю. У елементах, що відмовилися під час ремонту нових відмов не виникло; при відмові системи справні основні елементи не відмовляють.

У разі, коли $0 \ll 1/\mu$, параметр контролю основних елементів x включає лише середнє напрацювання до відмови τ, а не на k; при цьому для практичних розрахунків можна покласти $\tau = 0$.

Система з k основних та одного резервного елемента [25]. Припускається, що разом можна відновити до m елементів (m=1,2 число числом ремонтних бригад (установок)). Визначення показників надійності такої системи необхідного для аналізу:

$$T^{(0)} = T^{(1)} + x \,/\, [\mu(1 - v + A_1 + A_2)]$$
$$T^{(1)} = (1 + A_1 + A_2)/[k\ \ (1 - x + A_1 + A_2)]$$
$$K = (1 + A_1 + A_2)[1 + A_1 + A_2 + \ k(A_3 + A_4 + A_5)]$$
$$\tau = (A_3 + A_4 + A_5)/[\lambda(1 - x + A_1 + A_2)]\,, \qquad (3.1)$$

де $A_1 = \dfrac{k\ \ + \lambda_1}{\lambda}$

$$A_2 = \frac{(1 - \eta)\,\lambda_1\,\tau_k}{1 + \ \tau_k\,(k\ \ + \eta\,\lambda_1)}$$

$$A_3 = \frac{(1 - x)\lambda\theta}{1 - \ \lambda_1\tau}$$

$$A_4 = \frac{[1-x)\lambda A_2 + (1-\eta)\,\lambda_1 A_3]\tau_k\theta}{\tau_k + \tau + \eta\,\lambda_1\tau_k\theta}$$

$$A_5 = \frac{\lambda A_1 + \lambda A_2 + \lambda A_3}{m\text{«}} \qquad (3.2)$$

Також виділяється поширений частний випадок: дубльований пристрій ($k = 1$) з навантаженим резервом ($\lambda_1 = \lambda$), періодичний контроль резерву відсуває ($\tau_k = \infty$):

$$T^{(1)} = 1/[\lambda\left(1 - (1+\eta)\left(\frac{x}{2} - \frac{\lambda}{\mu}\right)\right)] \qquad (3.3)$$

4 ДОСЛІДЖЕННЯ МЕТОДУ СТВОРЕННЯ ВІДМОВИСТОЙЛИВИХ ВІРТУАЛЬНИХ СИСТЕМ

4.1 Планування досліджень

Перед описом самих методів дослідження запропонованого методу Відмовостійкі багатошарові віртуальні інфраструктури в хмарному середовищі враховується, що експерименти моделюють ситуації виходу з ладу окремих вузлів, такі проблеми як вихід з ладу ряд серверів через проблеми електропостачання не розглядається.

Для досліджень за допомогою інструменту Docker розгорнуто систему з вкладеною віртуалізацією та реалізовано у хмарі, побудованій на платформі Google Cloud Platform в ізольованому сегменті інфокомунікаційної мережі. Захист програм на Google Cloud Platform (GCP) – це практика спільної відповідальності (рисунок 4.1).

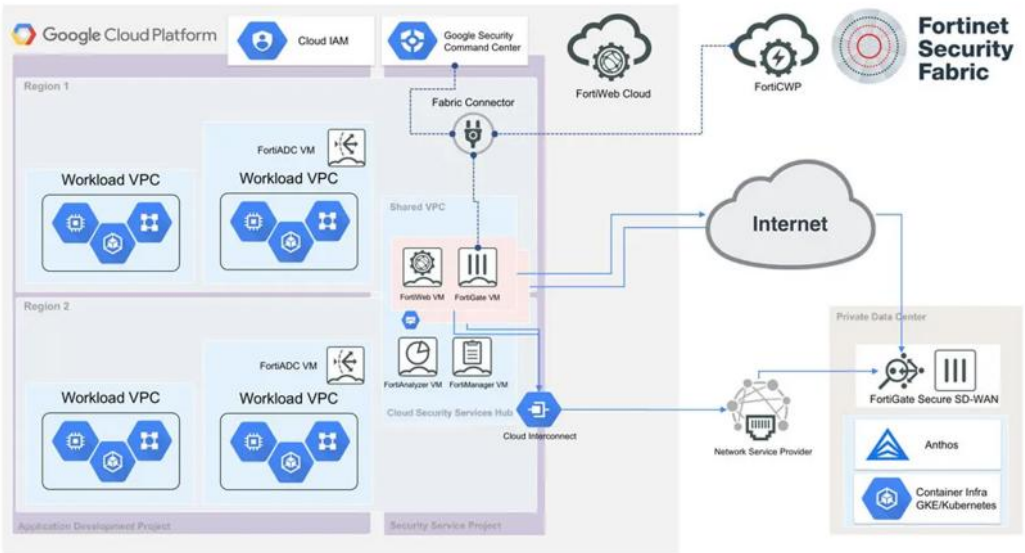

Рисунок 4.1 – Архітектура Google Cloud Platform

Хоча базова інфраструктура забезпечена Google, користувачі послуг хмарних обчислень несуть відповідальність за захист даних та програм, які вони створюють на хмарній інфраструктурі. Система Fortinet Security Fabric для GCP пропонує організаціям можливість впевнено розгортати програми на GCP або в кількох хмарах та центрах обробки даних за збереження високого рівня безпеки.

У цьому випадку штучно емітували мережеві атаки на хмарний міжмережевий екран. Щоб прискорити вихід із ладу вузлів хмарного міжмережевого екрану, у ньому реалізовані примітивні функції фільтрації вхідного трафіку. Служба, яку має захищати хмарний брандмауер, є системою на платформі Apache Server, яка також розміщується в хмарі.

У дослідженнях не розглянуті варіанти проблеми збою інфраструктури загалом через технічні проблеми вендора, т.к. в цьому випадку користувачеві потрібно лише повідомити про проблеми технічної підтримки сервісу, що використовується, і дочекатися детальної інформації про можливі рішення на стороні сервера або користувача. Віртуальні машини мають такі характеристики: 2 ядра на 1 віртуальний процесор із підтримкою технології Intel Turbo Boost 3.0, 8 Гб ОЗУ та віртуальний жорсткий носій ємністю 1 Тб.

4.2 Імітація падіння вузла

Для імітації падіння вузла проведемо 2 варіанти: при першому варіанті вузол відмовить без можливості відновлення (тобто пошкодження апаратної частини), при другому варіанті вузол інфраструктури можна відновити шляхом перезапуску контейнера.

У ході виконання першого варіанта вручну було прописано додаткові конфігураційні налаштування, що передають операційній системі віртуальних машин повідомлення про несправну шину пам’яті. Шина пам’яті призначена передачі інформації між процесором і основний пам’яттю системи. Ця шина з’єднана із північним мостом або мікросхемою Memory Controller Hub. Залежно від типу пам’яті, що використовується набором мікросхем (а отже, і системною

платою), шина пам’яті може працювати з різними швидкостями.

Система розпізнала проблему через 6,5 секунди після застосування налаштувань конфігурації. Оскільки код помилки зарезервований під апаратні неполадки, система не робила перезапуск контейнера, а відразу передала сигнал на балансувальник навантажень інформацію про повне падіння вузла з метою розподіл процесорного часу між вкладеними віртуальними машинами. При цьому ефективність всього хмарного сервера залежить від кількості виділеного часу, оскільки брак ресурсів призводить до виникнення черг із запитів, що обробляються у віртуальних машинах, та затримки в обслуговуванні вузлів інфраструктури. Даний хід системи дозволив заощадити час, який система не змогла витратити на безуспішну спробу перезапуску контейнера. а відразу розпочала обробку завдань на доступних вузлах використовуючи підвищене значення тактової частоти центрального процесора. Додатково обслуговуючий персонал інфраструктури отримав оповіщення про критичний збій у роботі системи.

У ході виконання другого варіанту було проведено імітацію "зависання" однієї з віртуальної машини шляхом запуску програми-демона на її потужностях. Через 20 секунд після захоплення віртуальної машини шкідливим програмним забезпеченням спрацював тригер на системі контролю версій GitLab, який дав команду системі про вимушений перезапуск контейнера. Перезапуск контейнера із вкладеною віртуалізацією зайняв 95 секунд. Після тригер надіслав повторний запит для отримання відомостей про стан робочої станції та отримав відповідь про стабільну роботу. Повний процес виявлення, перезапуск та контроль роботи після аварійного перезапуску зайняв близько трьох хвилин. Додатково до журналу внесено інформацію про аварійну ситуацію.

4.3 Імітація DDos-атаки

Розподілена атака типу «відмова в обслуговуванні», або DDoS – це справжнісіньке бомбардування центрального сервера одночасними запитами

даних. Зловмисник надсилає такі запити з кількох зламаних систем.

DDoS-атака може бути для зловмисника способом вимагання грошей. Крім того, подібні атаки іноді приносять переваги конкурентам (якщо спрямовані проти комерційної організації) чи політичну вигоду уряду чи «хактивістам». Збій у корпоративній мережі будь-якої компанії може стати серйозним успіхом для багатьох людей.

Тривалість DDoS-атак може змінюватись в залежності від різних факторів. Такі, як Ping of Death, зазвичай блискавичні. Атаки типу Slowloris відбуваються довше. Згідно з звітом Radware, 33% DDoS-атак тривають близько години, 60% тривають майже добу, а 15% не припиняються протягом місяця.

Захист бізнесу від DDoS-атак – це важлива частина стратегії забезпечення безпеки мережі. Щоб укрити мережу від атак по безлічі напрямків, необхідно розгорнути комплексне та цілісне IT-середовище, компоненти якого працюватимуть як єдине ціле.

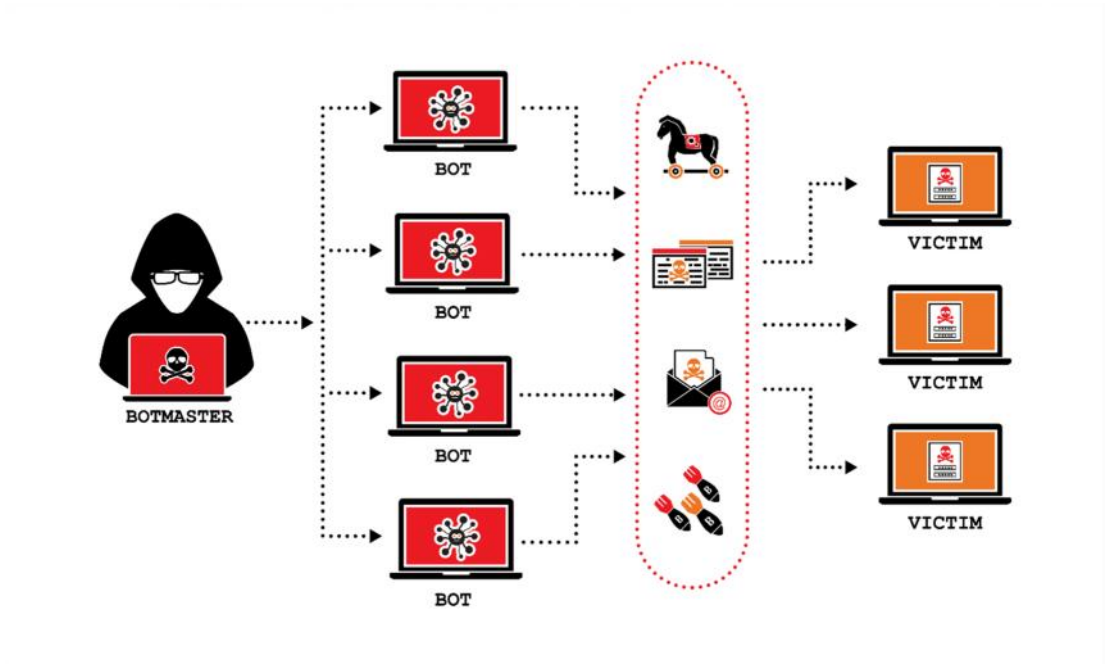

Рисунок 4.2 – DDoS-атака

Одним із варіантів захисту від DDoS-атак є використання міжмережевих

екранів (брандмауерів) нового покоління. Міжмережевий екран може бути окремою програмою, а може входити до складу будь-якої програми. Багато сучасні антивірусні програми включають Міжмережевий екран як компонент, що захищає комп'ютер. Іноді міжмережевий екран виконаний у вигляді програмно-апаратного комплексу.

Для захисту брандмауера слідкує за параметрами вхідного та вихідного трафіку. Класичні брандмауери, так звані пакетні фільтри, оцінюють трафік за параметрами мережевого рівня і приймають рішення про те, чи пропускати або не пропускати кожен IP-пакет, за його властивостями.

Таким чином він намагається повністю зайняти інтернет-канал та виснажити ресурси ОЗУ компанії-жертви. Кінцева мета полягає у виведенні з ладу систем такої компанії та перериванні її бізнес-процесів.

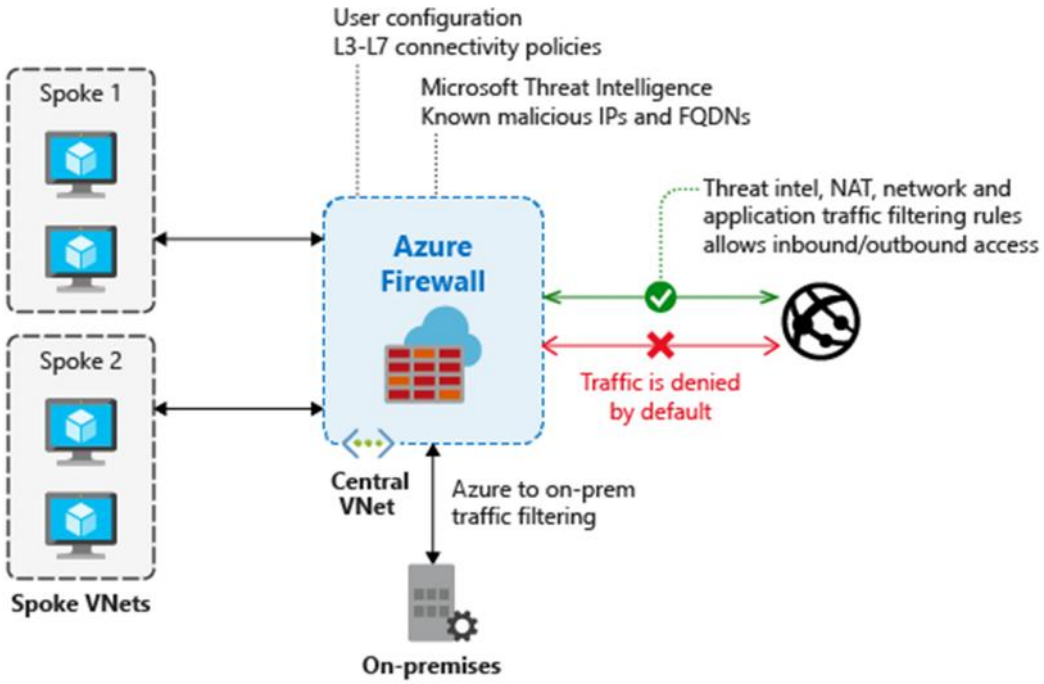

Рисунок 4.3 – Принцип роботи брандмауера

У випробуваній системі хмарний міжмережевий екран був хмарний сервер (рішення IaaS), на якому було розгорнуто набір віртуальних машин з використанням технології PaaS, на якому були реалізовані окремі функції

міжхмарного хмарного екрану за принципом SaaS. Мережеву інфраструктуру в хмарному сервері було реалізовано через віртуальні комутатори.

Атака міжмережевого екрану була класичною атакою контрабанди HTTP-запитів. Перший хмарний брандмауер відмовив в обслуговуванні на 12 секунді. Час відновлення зайняв 40 секунд. Другий хмарний брандмауер відмовив в обслуговуванні на 10 секунді, але час відновлення зайняв 30 секунд.

Друга атака була низькорівневою DDoS-атакою на сервери віртуалізації (включаючи оркестратор) хмарної інфраструктури з метою переведення її в стан відмови в обслуговуванні. У цьому випадку хмарний брандмауер не працював, поки не спрацював опитування вузлів і тригер дав команду перезапуску контейнера.

4.4 Імітація міграції

Також запропоноване рішення має можливість перенести систему на потужності іншого вендора. Для цього від системного адміністратора потрібно лише дати інформацію про нову фізичну адресу.

У ході дослідження знадобилося 5200 секунд для розгортання міжхмарного хмарного екрану в умовній хмарі іншого провайдера і повторної прив’язки IP-адрес до домену за допомогою служби динамічного DNS.

Таблиця 4.1 – Результати досліджень затрат часу на усунення уражень системи

Тип ураження	Час усунення	Потрібне втручання
Ураження вузла, що не потребує фізичної заміни	до 5 хвилин	Ні
Ураження вузла, що потребує фізичної заміни	до 5 хвилин	Так
DDoS-атака	40 секунд	Ні
Міграція серверу	5200 секунд	Так

Таким чином запропоноване рішення має змогу відновлювати робочий стан в напівавтоматичному режимі. Втручання людини потрібно лише, коли фізичні компоненти потребують заміни або внаслідку падіння серверних потужностей вендора система потребує міграції, тоді єдине, що потребується від людини – вказати нову фізичну адресу.

ВИСНОВКИ

В кваліфікаційній роботі роботи поставлена і успішно вирішена задача створення відмовостійких багатошарових віртуальних інфраструктур в хмарному середовищі, а також проведений аналіз впливу окремих її вузлів та можливість їх автоматичного відновлення.

Зокрема:

- проведене моделювання, що доказує ефективність запропонованого рішення досягає максимального ефекту серед існуючих рішень;

- запропоноване рішення, яке дозволяє системі віртуальної інфраструктури відновлювати працездатність у автоматичному або напів-автоматичному режимах;

- розроблена математична модель для розрахунку часу відновлення системи, яка дозволяє отримати максимально близьке значення до реальної моделі;

- в основі рішення використовуються лише платформи, які можуть застосовуватись на основі free-use, тобто безкоштовно обмежуювачи користувача у функціоналі, що не є критичним для конкретного рішення;

- запропоноване рішення є універсальним та має змогу працювати з усіма вендорами віртуальних сервісів, а також може бути застосовано за допомогою інших сервісів безперервного розгортання.

Головною задачею роботи було знайти метод відмовостійокстi віртуальних систем, яка б не потребувала додаткових потужностей та не призводила до використання основних потужностей, що призвело б до падіння продуктивності системи загалом.

У роботі запропоновано використання сучасних засобів безперервного розгортання та розробки, що використовуються більшістю компаній. Дане рішення дозволяє не тільки зробити систему стійкою до відмови, а також захистити від різного виду атак шляхом міграції з однієї фізичної адреси на іншу,

в доступному пулі виділеним системним адміністратором раніше.

У роботі розглянуті різноманітні варіанти падіння системи віртуальних інфраструктур такі, як: відмова системи через неполадки на стороні вендора хмарних сервісів, через виходу з ладу фізичних компонентів серверної інфраструктури, хакерські атаки, а також сценарії, коли потрібно перенести інфраструктуру віртуальної системи на інший вендор.

Новизна даної кваліфікаційної роботи полягає в тому, що вперше запропоновано метод, який здійснює весь цикл із забезпечення роботи віртуальної інфраструктури: від розгортання її на базі вендора хмарних сервісів до підтримки її безвідмовної роботи. Вперше для забезпечення розгортання системи та її відмовостійкості запропоновано використовувати сервіси безперервного розгортання та розробки. Дані сервіси дають можливість використовувати програмовані тригери, які активуватимуть різні сценарії роботи. Таким чином, у запропонованому рішенні використовується тригер, який повідомляється про вимушений перезапуск контейнера. Після успішного перезапуску контейнера, що займає близько 5 хвилин системи середнього масштабу, система відновлює свою працездатність.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Vitalii Tkachov, Volodymyr Tokariev, Iryna Ilina, and Stanislav Partyka, "Modified Traveling Salesman Problem for a Group of Intelligent Mobile Objects and Method for Its Solving," International Journal of Electrical and Electronic Engineering & Telecommunications, Vol. 10, No. 1, pp. 1-7, January 2021. Doi: 10.18178/ijeetc.10.1.1-7.

2. Ткачов В.М. Разработка алгоритма мультиагентного управления группой мобильных «s-bot» / В.М. Ткачов, В.В. Токарєв, Г.І. Чурюмов // Реєстрація, зберігання і обробка даних. – Київ: Інститут проблем реєстрації інформації НАН України, 2019. – № 1. – Т. 21. – С. 46-56.

3. Ткачов В.М. Метод передачі даних в комп’ютерній мережі проміжного зберігання даних складної інформаційної системи / В.М. Ткачов // Системи управління, навігації та зв’язку. – Полтава: Полтавський національний технічний університет ім. Ю. Кондратюка, 2017. – № 3 (43). – С. 117-119.

4. Ткачов В.М. Проблема передачі даних типу Big Data у мобільній системі «мультикоптер-сенсорна система» / В.М. Ткачов, В.В. Токарєв, В.О. Радченко, В.О. Лебедєв // Системи управління, навігації та зв’язку. – Полтава: Полтавський національний технічний університет ім. Ю. Кондратюка, 2017. – № 2 (42). – С. 154-157.

5. Vitalii Tkachov, Anna Budko, Kateryna Hvozdetska and Daryna Hrebeniuk. Method of Building Dynamic Multi-hop VPN Chains for Ensuring Security of Terminal Access Systems // IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T): Kharkiv 06-09 oct. 2020, Kharkiv.

6. Ткачов В.М. Аналіз методів забезпечення відмовостійкості оверлейних мереж / В.М. Ткачов, К.П. Гвоздецька // Проблеми інформатизації : тези доп. 8-ї міжнар. наук.-техн. конф., 26-27 листопада 2020 64 р., м. Черкаси, м. Харків, м. Баку, м. Бельсько-Бяла. Т. 1 / Черк. держ. технолог. ун-т [та ін.]. – Харків, 2020.

– С. 44.

7. Tkachov, V., Bondarenko, M., Ulyanov, O., & Reznichenko, O. (2019, December). Overlay Network Infrastructure for Remote Control of Radio Astronomy Observatory. In 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT) (pp. 161-165).

8. Tkachov V.M. Architecture of Overlay Network with Nested VPN Tunneling / V. Tkachov, M. Bondarenko, M. Hunko // Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: матеріали десятої міжнародної науково-технічної конференції. – Баку: ВА ЗС АР; Харків: ДП «ХНДІ ТМ»; Жиліна: УмЖ, 2020. – С. 36.

9. Tkachov V. Technology of Load Balancing in Anonymous Network Based on Proxy Nodes Cascade Platform / V. Tkachov, M. Hunko, M. Bondarenko, S. Artyomov // Четверта міжнародна науково-технічна конференція «Комп’ютерні та інформаційні системи і технології». Збірка наукових праць. Харків: ХНУРЕ. – 2020. – С. 82.

10. Ткачов В.М. Програмний кластер для паралельної обробки великих обсягів даних / В.М. Ткачов, Ю.А. Кривобоков, К.П. Гвоздецька // Міжнародна наукова інтернет-конференція "Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 49)" / Збірник тез доповідей: випуск 19 (м. Тернопіль, 10 червня 2020 р.). – Тернопіль. – 2020. – 31-33 с.

11. Ткачов В.М. Застосування технології OpenVPN в рамках сервісу «Health Tracker» / В.М. Ткачов, А.О. Карасьов // 73-я науково-технічна конференція професорсько-викладацького складу, науковців, аспірантів та студентів ОНАЗ ім. О.С. Попова, 12-14 грудня 2018 року. – Одеса. – 2018. – С. 157-158.

12. Ткачев В.Н. Применение метода предотвращения коллизий при параллельной обработке данных в полинговых сетях контроля состояния сложных распределенных систем / В.Н. Ткачев, А,А. Коваленко, В.О. 65 Лебедев // Третя міжнародна науково-технічна конференція «Проблеми інформатизації» 12-13 листопада 2015 року. – Черкаси–Баку–Бельсько-Бяла– Полтава. – 46 с.

13. Коваленко А. А., Кучук Г. А. Оптимальное управление трафиком мультисервисной сети на основе методов последовательного улучшения решений //Системи озброєння і військова техніка. – 2016. – №. 3. – С. 59-63.

14. Кринкин К.В. Создание алгоритмов маршрутизации в динамических компьютерных сетях с использованием неполных данных / Кринкин К.В. – Санкт-Петербург, 2014 г. – 147 с.

15. Ткачов В.М. Питання кібербезпеки при впровадженні IAAS-рішень хмарних вендорів / В.М. Ткачов, С.О. Партика // Проблеми науковотехнічного та правового забезпечення кібербезпеки у сучасному світі. – Матеріали першої міжнародної науково-практичної конференції. – Харків, НТУ «ХПІ». – 2016. – с. 38.

16. Tkachov V.M. Providing information security in systems of business process management in the IAAS-vendor environment / V.M. Tkachov, S.O. Partyka, V.O. Lebediev // Сучасні напрями розвитку інформаційнокомунікаційних технологій та засобів управління: матеріали шостої міжнародної науково-технічної конференції. – Полтава: ПНТУ; Баку: ВА ЗС АР; Кіровоград: КЛА НАУ; Харків: ДП «ХНДІ ТМ», 2016. – С. 25.

17. Ткачев В.Н. Метод передачи больших массивов данных с использованием SaaS-маршрутизаторов в оверлейных сетях / В.Н. Ткачев, В.О. Лебедев // Перша міжнародна науково-технічна конференція «Комп’ютерні та інформаційні системи і технології». Збірка наукових праць. Харків: ХНУРЕ. – 2017. – С. 31.

18. История сетей [Електронний ресурс] – Режим доступу : www/ URL: https://refdb.ru/look/1986198.html – 28.05.2020 р. – Загол. з екрану.

19. Bondarenko M.E., Pugach A.V. Scheme of big data virtualization in anonymous computer networks // Міжнародна наукова інтернет-конференція «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення» (випуск 48), 12 травня 2020 року – У друці.

20. Чайковская А. С., Кудрявцев К. Я., Павлов Е. К. Представление больших наборов динамически изменяющихся удаленных данных в

приложениях WPF //International Journal of Open Information Technologies. – 2017. – Т. 5. – №. 9.

21. Авдошин С. М., Лазаренко А. В. Технология анонимных сетей //Информационные технологии. – 2016. – Т. 22. – №. 4. – С. 284-291.

22. Mekky H. et al. Network function virtualization enablement within SDN data plane //IEEE INFOCOM 2017-IEEE Conference on Computer Communications. – IEEE, 2017. – С. 1-9. 5. Babiceanu R. F., Seker R. Big Data and virtualization for manufacturing cyber-physical systems: A survey of the current status and future outlook //Computers in Industry. – 2016. – Т. 81. – С. 128-137.

23. Маркин Д. О., Архипов П. А., Галкин А. С. Исследование устойчивости анонимной сети на основе технологий веб-прокси //Вопросы кибербезопасности. – 2016. – №. 2 (15).

24. Трофимов В. В., Кияев В. И., Газуль С. М. Использование виртуализации и контейнерной технологии для формирования информационной инфраструктуры //Материалы XX Международной конференции по мягким вычислениям и измерениям (SCM-2017). Санкт- Петербург, ЛЭТИ. – 2017. – С. 348-352.

25. Wolfert S. et al. Big data in smart farming–a review //Agricultural 62 Systems. – 2017. – Т. 153. – С. 69-80.