Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інфокомунікацій
(повна назва)
Кафедра Інфокомунікаційної інженерії імені В.В. Поповського
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Розробка та дослідження системи моніторингу мережі віддаленого доступу
(тема)

Виконав:
студент 2 курсу, групи ІКІм-21-1
Колтаков О.А.
(прізвище, ініціали)

Спеціальність: 172 Телекомунікації і радіотехніка
(код і повна назва спеціальності)

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

Освітня програма: Інфокомунікаційна інженерія
(повна назва освітньої програми)

Керівник: доцент кафедри ІКІ ім. В.В. Поповського
Токар Л.О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Лемешко О.В.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інфокомунікацій
(повна назва)
Кафедра Інфокомунікаційної інженерії імені В.В. Поповського
(повна назва)
Рівень вищої освіти другий (магістерський)
Спеціальність 172 Телекомунікації і радіотехніка
(код і повна назва)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Інфокомунікаційна інженерія
(повна назва)

ЗАТВЕРДЖУЮ
Зав. кафедри_____
(підпис)
«____» _____2022р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Колтакову Олександру Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: Розробка та дослідження системи моніторингу мережі віддаленого доступу
затверджена наказом по університету від «24» жовтня 2022р. № 1389 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 22.12.2022р.
3. Вихідні дані до роботи: методи та принципи побудови мережі Call - центру, вимоги до побудови систем віртуалізації з використанням гіпервізора VMWare ESXI, вимоги до побудови систем моніторингу телефонної мережі з використанням Zabbix серверу, моделювання та налаштування кластеру серверів на основі Asterisk, методика дослідження кластеру серверів з балансуванням навантаження
4. Перелік питань, що потрібно опрацювати в роботі:
1) Аналіз особливостей функціонування та реалізації мережі Call - центру
2) Налаштування тестового стенду Call – центру. Керування балансуванням навантаження викликів між серверами Asterisk
3) Розробка системи моніторингу з використанням Zabbix сервера
4) Кластерізація Call Server. Дослідження роботи одиночного CS у кластері
5) Дослідження характеристик CS у кластері

5. Перелік графічного матеріалу із зазначенням креслень, плакатів, комп'ютерних ілюстрацій: Демонстраційний матеріал у вигляді ppt-презентації.

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		(підпис)	(дата)
Основна частина	доцент Токар Любов Олександровна		20.12.22 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	01.09.2022	Виконано
2	Збір матеріалів для дослідження	22.09. 2022	Виконано
3	Розробка 1 розділу	16.10.2022	Виконано
4	Розробка 2 розділу	12.11.2022	Виконано
5	Розробка 3 розділу	20.11.2022	Виконано
6	Розробка 4 розділу	01.12.2022	Виконано
7	Оформлення кваліфікаційної роботи	10.12.2022	Виконано

Дата видачі завдання 01 вересня 2022 року

Студент _____ Колтаков О.А.
(підпис) (прізвище, ініціали)

Керівник роботи _____ доцент Токар Л. О.
(підпис) (посада, прізвище, ініціали)

Робота не містить відомостей, заборонених до відкритого публікування

Студент Олександр КОЛТАКОВ

Керівник Любов ТОКАР

РЕФЕРАТ

Пояснювальна записка: 91 с., 51 рис., 7 табл., 40 джерел, 3 додатка.

СЕРВЕР, КЛАСТЕР, БАЛАНСУВАННЯ НАВАНТАЖЕННЯ, ГОЛОСОВИЙ ВИКЛИК, МОНІТОРИНГ, ПЕРЕАДРЕСАЦІЯ, CALL-ЦЕНТР

Об’єкт дослідження – процеси розробки та проєктування системи моніторингу кластерів серверів викликів.

Предмет дослідження - методи і засоби покращення функціонування кластерів серверів викликів.

Мета роботи – розробка системи моніторингу кластерів серверів викликів, дослідження та аналіз параметрів кластерів серверів викликів для забезпечення ефективної роботи та необхідної відмовостійкості.

Методи досліджень – аналіз, формалізація, порівняння.

В роботі розроблено та налаштовано систему моніторингу для дослідження кластеру серверів викликів з використанням open source продукту Zabbix.

Досліджено динамічні політики переадресації для оцінки їх впливу на продуктивність кластера серверів викликів при різних поєднаннях параметрів робочого навантаження та за різних сценаріїв перевантаження.

Результати роботи можуть бути корисними виробникам телекомунікаційного обладнання, постачальникам телекомунікаційних послуг, а також представляти інтерес у контексті інших кластерних систем, що обробляють робоче навантаження, орієнтоване на транзакції.

ABSTRACT

Explanatory note: 91 p., 51 fig., 7 table, 40 sources, 3 add.

SERVER, CLUSTER, LOAD BALANCING, VOICE CALL, MONITORING, FORWARDING, CALL CENTER

The object of research is the processes of development and design of a system for monitoring clusters of call servers.

The subject of research is methods and means of improving the functioning of clusters of call servers.

The purpose of the work is to develop a system for monitoring clusters of call servers, research and analysis of parameters of clusters of call servers to ensure efficient operation and the necessary fault tolerance.

Research methods - analysis, formalization, comparison.

The work developed and configured a monitoring system for researching a cluster of call servers using the open source Zabbix product.

Dynamic forwarding policies are investigated to evaluate their impact on the performance of a cluster of call servers under different combinations of workload parameters and under different overload scenarios.

The results of the work may be useful to telecommunication equipment manufacturers, telecommunication service providers, and also be of interest in the context of other cluster systems processing transaction-oriented workloads.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

АТС - автоматична телефонна станція
ВМ – віртуальна машина
СМТ - система моніторинга трафіку
ОС - операційна система
ЦП - центральний процесор
CRM - customer relationship management
CPU - central processing unit
CS - call server
DDoS - distributed denial-of-service
IP - internet protocol
IETF - internet engineering task force
IOP - input/output processor
ISDN - integrated services digital network
ITU-T - International Telecommunication Union - Telecommunication sector
FIFO - first in, first out
KVM - kernel-based virtual machine
MCU - multipoint control unit
MGCP - media gateway control protocol
MPLS - multiprotocol label switching
NAT - network address translation
ODI - origination delay improvement
OSI - open systems interconnection model
OAM - operations and management
PCI- peripheral component interconnect
PBX - private branch exhange
PHP - personal home page
PE - processing element
POD - percentile origination delay
PSTN - public switched telephone network
QoS - quality of service
RSVP - resource reservation protocol

RTP - real-time transport protocol
SDP - session description protocol
SIP - session initiation protocol
SMS - short message service
SRTP - secure real transport protocol
TCP - transmission control protocol
UDP - user datagram protocol
VoIP - voice over IP
XML - extensible markup language

ВСТУП

Із зростанням попиту на телекомунікаційні послуги сервери голосових викликів та викликів даних повинні забезпечувати більшу пропускну здатність обробки викликів. Інтернет та мобільні пристрої сприяли збільшенню потреб в комутаційних потужностях телекомунікаційних серверів. Через ці вимогливі програми можливості існуючих серверів обробки викликів доведено до краю. Щоб впоратися із зростаючим попитом на пропускну здатність, забезпечити необхідну відмовостійкість та мінімізувати експлуатаційні витрати для телефонних компаній, необхідні дослідження кластеризації серверів викликів. Такі кластери повинні працювати як з мережами з комутацією каналів, так і з мережами передачі голосу по IP (internet protocol).

Уникаючи єдиної точки відмови, кластер серверів викликів може вирішити характерні проблеми безпеки. Забезпечуючи географічну надмірність, збій сервера обробки викликів через терористичну атаку або стихійне лихо може ефективно оброблятися іншими серверами обробки викликів у кластері, які можуть розділити робоче навантаження сервера обробки викликів.

Мета роботи - розробка системи моніторингу кластерів серверів викликів, дослідження та аналіз параметрів кластерів серверів викликів для забезпечення ефективної роботи та необхідної відмовостійкості.

В кваліфікаційній роботі розроблено та налаштовано систему моніторингу з використанням open source продукту Zabbix для дослідження кластеру серверів викликів на основі Asterisk. Проведено налаштування тестового стенду Call-центру. Проведено дослідження динамічних політик переадресації та характеристик кластерів серверів обробки викликів з метою оцінки їх впливу на продуктивність системи за різних поєднань параметрів робочого навантаження, а також за різних сценаріїв перевантаження.

Окремі результати роботи доповідались на трьох Міжнародних наукових конференціях [1-3].

1 АНАЛІЗ ОСОБЛИВОСТЕЙ ФУНКЦІОНУВАННЯ ТА РЕАЛІЗАЦІЇ МЕРЕЖІ CALL-ЦЕНТРУ

1.1 IP - телефонія як сучасне рішення для зв’язку

На сьогоднішній день можливості IP-телефонії багато в чому перевершують функціонал аналогових систем зв'язку. На відміну від аналогових телефонних систем зв’язку, сигнал в VoIP (voice over IP) мережах передається в цифровому вигляді. Це дає змогу використовувати пакетну комутацію та зменшити вимоги до смуги пропускання.

Такі сучасні програмні продукти, як VMware або KVM (kernel-based virtual machine), дають змогу віртуалізувати будь яку систему. Віртуалізація цифрової АТС (автоматична телефонна станція) дає змогу економити обладнання, спрощувати його адміністрування. Віртуальні АТС можуть забезпечувати понад 100 додаткових функцій. Сюди належать: функція переадресації дзвінків, голосові привітання, голосова пошта, запис розмов, зберігання розмов та ін.

Замовляючи послуги IP-телефонії, керівник компанії може бути впевнений, що співробітники зможуть швидко додзвонитися до іншого співробітника або конкретного відділу. Цього вдається досягти завдяки багатоканальним номерам.

Сьогодні ж телефонні мережі та Інтернет поєднуються в єдину інфраструктуру. Голосовий IP трафік вже потіснив звичну телефонію й конкурує зі стільниковим зв'язком. З розвитком IP-телефонії збільшується її функціонал, і до відсутності роумінгу в мережі Інтернет додається безкоштовний аудіо-відео зв'язок, можливість проведення конференцій, хмарні послуги тощо.

Однією з важливіших переваг IP-телефонії є [4]:

- велика різноманітність сервісів;
- високий рівень мобільності;
- інтеграція із різними CRM (customer relationship management) системами;
- легке та швидке масштабування.

IP-телефонія будується з урахуванням кількох стандартизованих протоколів зв'язку. Основним протоколом є стандарт H.323 від ITU (International Telecommunication Union), за яким працюють транснаціональні оператори Інтернет-телефонії. Управління викликами в протоколі H.323 здійснює gatekeeper (контролер зони) – програма або пристрій, що знаходиться у провайдера IP-

телефонії, що координує всіх клієнтів, які об'єднані у спільну зону. Під його керуванням знаходяться термінали, шлюзи, пристрої керування конференціями та ін. Завдяки шлюзам у протоколі H.323 можливе об'єднання телефонної та IP-мереж [5]. Так, шлюзом може бути адаптер для аналогових телефонів (наприклад, GXW), який підключається до Інтернету та аналогового телефону.

Більш гнучким й масштабованим вважається протокол SIP (session initiation protocol), який використовується майже всіма провайдерами IP-телефонії, що дозволяє не лише розмовляти через Інтернет, а й обмінюватися відео, миттєвими повідомленнями, грати в онлайн-ігри тощо. Цей протокол регламентує лише процедуру встановлення з'єднання між пристроями, а передачі даних підключається мережевий протокол передачі потокових даних SDP (session description protocol). У SIP більше функцій, ніж у H.323, і ця різниця пояснюється тим, що цей протокол розроблявся не ITU, а відкритою групою міжнародного співтовариства проєктувальників, вчених, мережевих операторів і провайдерів IETF (Internet Engineering Task Force). Для роботи SIP-протоколу потрібні проксі-сервери, які встановлені у провайдерів IP-телефонії, і термінали користувачів. Ними можуть виступати IP-телефони, шлюзи, а також комп'ютери або смартфони зі встановленим програмним забезпеченням (софтфоном) з підключенням до Інтернету [6].

За якість передачі голосу у всіх протоколах відповідають кодеки – алгоритми перетворення голосу у необхідний цифровий формат. В IP-телефонії голос перетворюється на так звані пакети, які передаються по Інтернет-мережах. До найбільш популярних кодеків відносять: G.711, G.722, GSM.

1.2 Структура мережі Call-центру

Call-центр являє собою центр обробки телефонних викликів, принцип побудови якого ґрунтується на маршрутизації викликів агентів за певними правилами, які розробляються в компанії та дозволяють якісно та ефективно обслуговувати клієнтів, а також підтримувати черги дзвінків. Він здатний не тільки приймати та обробляти запити, але й використовувати для контактів з клієнтами звичайну пошту, факсимільний та мобільний зв'язок, Інтернет, SMS (short message service) тощо.

Основні вимоги щодо правильного та стабільного функціонування Call-центру наступні [7]:

- відмовостійкість системи;

- якісний та швидкий доступ до мережі Інтернет;

- використання системи, яка здатна інтегрувати в собі декілька технологій (наприклад, передача даних, голосу, відео);

- безпека дзвінків шляхом шифрування даних;

- масштабованість;

- здійснення дзвінків в загальну абонентську мережу PSTN (public switched telephone network).

На рис. 1.1 представлено схему організації Call-центру компанії.

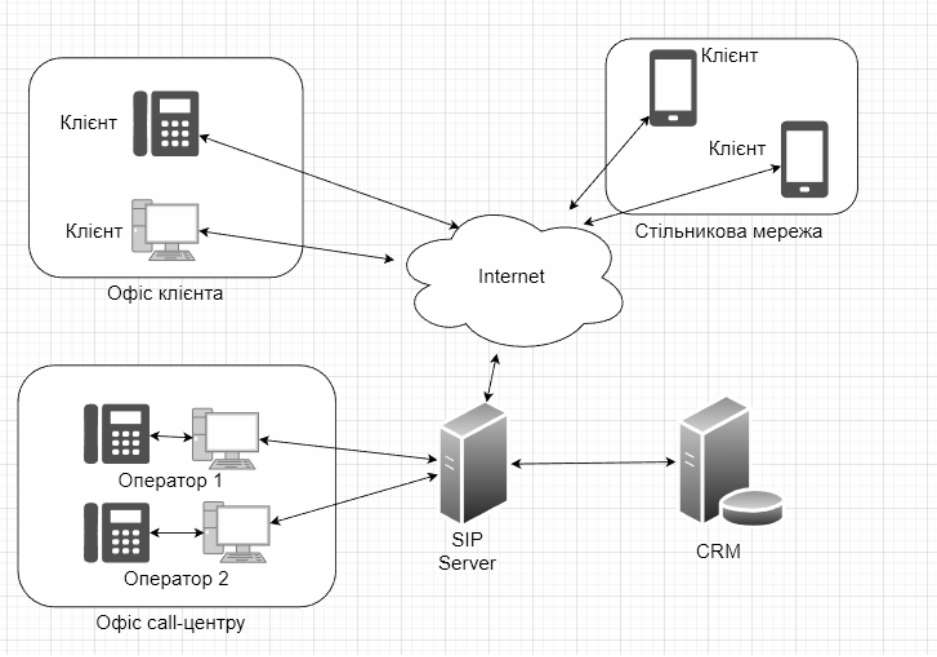

Рисунок 1.1 - Схема організації Call-центру компанії

Основним елементом мережі є SIP сервер, або кластер з декількох серверів (якщо компанія здійснює досить велику кількість дзвінків). Сервер може бути як фізичний, так і віртуальний (орендована віртуальна машина (ВМ). В якості програмного забезпечення може бути, наприклад, Asterisk. SIP сервер зв’язується з CRM системою, яка призначена для автоматизації стратегій взаємодії із замовниками (клієнтами). Разом вони формують ядро системи, яке має швидкісне з’єднання з мережею Інтернет, де SIP серверу виділяється номер, який виділено SIP провайдером. Офіс Call-центру з’єднаний з ядром та складається з IP-телефонів та комп’ютерів. Він може територіально знаходитись як в одному будинку з ядром мережі, так і в іншому місці.

1.3 Хмарне рішення реалізації АТС

В Україні на сьогоднішній день кожний крупний Інтернет-провайдер має в списку своїх послуг надання віртуальної АТС клієнту. Наприклад, такі провайдери, як Datagroupe, Kyivstar. Віртуальна АТС здатна виконувати функції традиційної телефонної станції. Тільки на відміну від традиційної телефонії, хмарна телефонія не потребує наявності прокладання проводів в офісі, достатня її наявність підключення до мережі Інтернет. Її головна перевага в тому, що телефонний номер не прив'язується до конкретного місця. Це означає, що можна в будь який час змінити офісне приміщення, місто або навіть країну без необхідності змінювати номер телефону на сайті своєї компанії. Така перевага надає можливість будувати роботу працівників віддалено без прив’язки до робочого місця. Хмарна телефонія дозволяє об’єднати в одну мережу телефонні номери різних операторів, що при правильному налаштуванні дозволить суттєво економити кошти на телефонні розмови. Для здійснення дзвінків клієнту не обов’язково купувати обладнання, він зможе дзвонити практично з будь-якого пристрою зв'язку: смартфону, комп’ютеру, ноутбуку, планшету та ін., попередньо встановивши відповідний SIP додаток, наприклад Zoiper або Sipnetic.

Віртуальна АТС необхідна для малого та середнього бізнесу. Порівнюючи її з апаратною АТС, можна виділити наступні переваги [8]:

- просте впровадження в діючу мережу;
- постійне оновлення системи АТС;
- хороша відмовостійкість, можливість резервування системи;
- можливість створення резервної копії системи, що дозволить повністю відновити систему за мінімальний проміжок часу;
- можливість легко інтегрувати її до CRM системи або конкретного додатку в системі.

1.4 Особливості процесу моделювання телефонної мережі

Моделювання та аналіз VoIP-мережі можна проводити у досить великій кількості симуляторів. Це можуть бути прості платформи, як Packet Tracer, GNS3, так і складні – EVE-NG, PNET або створення мереж в гіпервізорі.

Сам процес моделювання являє собою віртуалізації всієї системи, або деяких її частин. Це необхідно для того, щоб в подальшому уникнути проблем

при побудові реальної телефонної мережі, більш детально розібратися в деяких аспектах функціонування мережі, виявити слабкі ланки мережі, проаналізувати трафік тощо.

Загалом, побудова локальної телефонної IP-мережі в симуляторі аналогічна побудові комп'ютерної мережі. В якості SIP-клієнту або абонентських терміналів можна використати віртуальну машину Linux з встановленим у ній програмним забезпеченням Softphone. Наприклад, програми Ekiga, Zoiper, Ring та Twinkle. У кожній програмі при реєстрації SIP профілю, потрібно заповнити такі поля, як: ім'я користувача, домен, ім'я авторизації та пароль. Приклад налаштування в програмі Twinkle наведено на рис. 1.2 [4].

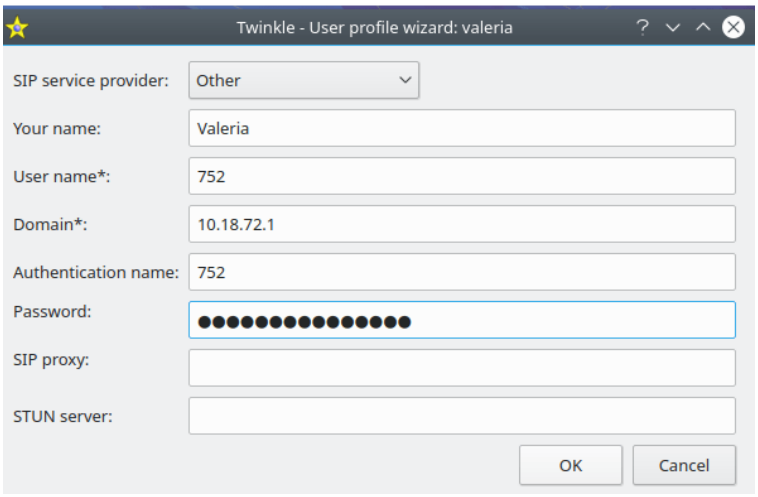

Рисунок 1.2 - Вікно налаштування SIP профілю програми Twinkle

Наступною особливістю є налаштування АТС, яка буде обслуговувати клієнтські пристрої. Для цього доцільно створити віртуальну машину з встановленою на ній віртуальною АТС, наприклад Asterisk, FreePBX або Elastic, налаштувати її в одну мережу з абонетськими пристроями, або маршрут до неї. Варто зазначити, що SIP-сервер сам не обробляє медіапотоки – це робить окремий медіа-сервер, використовуючи протокол RTP. У реалізаціях IP-АТС завжди SIP-сервер й медіа-сервер перебувають у одному фізичному сервері. Однак системи з великим навантаженням (наприклад, у великих VoIP-операторів) можуть використовувати медіа-сервер, встановлений на іншій фізичній машині, щоб краще справлятися з обробкою сесій. Також можливе розподілення навантаження на кілька медіа-серверів [4].

На мережевих пристроях, з метою покращення якості зв'язку, доцільно налаштувати приоритезацію VoIP-трафіку.

1.5 Переваги телефонії на базі Astersik

Офісну телефонну станцію називається PBX (private branch exhange). Вона забезпечує встановлення, підтримку та розрив з'єднань між апаратами, тобто комутацію. PBX дозволяє розділяти обмежені ресурси (міські лінії та номери) між необмеженою кількістю внутрішніх користувачів, за допомогою таких телефонних функцій, як внутрішній номерний план, переклад дзвінків, постановка на утримання та ін. Саме тому система PBX потрібна для будь-якої організації - вона дозволяє ефективно організувати телефонний зв'язок на підприємстві. PBX системи комутують пакети в мережі TCP/IP й називаються IP-PBX. IP-PBX працює на основі протоколів IP-телефонії. Також IP-PBX можуть підтримувати й традиційні лінії зв'язку - такі IP-PBX називаються гібридними.

Однією з таких PBX систем є проєкт Asterisk. В даний час Asterisk є найпопулярнішою відкритою IP-АТС у світі, займаючи майже 85 % ринку opens-ource продуктів. Модульна архітектура Asterisk дозволяє легко підключати в комутаційне поле будь-яку бізнес-логіку, написану практично будь-якою мовою програмування, або реалізовану власною мовою діалплану Asterisk. Нижче представлено список функцій Astersik [9]:

1) Підтримка як протоколів IP-телефонії, так й традиційних ліній зв'язку. У сервер з Asterisk можна вставити PCI (peripheral component interconnect) - плати Digium з аналоговими та/або цифровими портами в потрібній кількості та поєднанні.

2) Підтримка відеозв'язку.

3) Підтримка шифрування розмов.

4) Наявність простих й добре документованих інтерфейсів для інтеграції з іншими системами. Це дозволяє легко вбудовувати комунікації в бізнес-процеси та бізнес-додатки.

5) Наявність великих спільнот користувачів, розробників та інтеграторів.

6) Підтримка всіх базових та розширених функцій АТС: голосового меню, запису розмов, статистику дзвінків, музику на утриманні, голосову пошту, постановку дзвінків у чергу та розподіл за операторами.

7) Наявність готових дистрибутивів, що дозволяють розвернути на звичайному PC сервер IP-PBX за лічені хвилини.

Asterisk може працювати практично на будь-якій платформі Linux та на деяких інших ОС (операційних системах), таких як Solaris, BSD, MacOS X. CentOS найбільш популярний дистрибутив, який використовується з Asterisk.

1.6 Особливості моніторингу телефонної мережі

Сучасні мережі передачі проєктуються й будуються у вигляді реалізації різних концепцій, фундамент яких становить безліч телекомунікаційних протоколів: CAS, SS7, INAP, H.323, SIP тощо. Мережа IP-телефонії не виняток.

СМТ (система моніторингу трафіку) – це засіб, який покликаний здійснювати захоплення повідомлень, перерахованих вище (і не тільки) протоколів, і має набір зручних, інтуїтивно зрозумілих та інформативних інтерфейсів для його аналізу. Основне призначення СМТ – зробити сигнальні трасування та дампи за будь-який проміжок часу доступними для фахівців у будь-який час (у тому числі в режимі реального часу) без використання спеціалізованих програм (наприклад, Wireshark).

На відміну від звичайної мережі передачі даних, телефонні системи VoIP мають мінімальну стійкість до низької продуктивності, тому що навіть невелика затримка й втрата пакетів можуть мати руйнівний вплив на якість звуку. Слід зазначити, що система моніторингу трафіку VoIP не зовсім схожа за принципом моніторингу з класичною схемою, яка дозволяє складати карти мереж, контролювати доступність їх елементів, утилізацію ресурсів, периферію та багато іншого. Для цього в VoIP необхідно звести відповідний трафік з усіх пристроїв в одну точку – Capture Server. Таким чином, особливість моніторингу телефонної мережі виявляється у необхідності забезпечення централізації місця збору сигнального трафіку для подальшої обробки.

Інструменти телефонного моніторингу здатні:

- відстежувати та усувати неполадки якості дзвінків SIP VoIP;
- архівувати всі виклики, включаючи SIP, WebRTC, SKINNY RTP, SS7 через SCTP, T.38 та T.30 FAX (PDF) у базі даних CDR;
- декодувати та відтворювати дзвінки безпосередньо з графічного інтерфейсу користувача.

2 НАЛАШТУВАННЯ ТЕСТОВОГО СТЕНДУ CALL-ЦЕНТРУ. КЕРУВАННЯ БАЛАНСУВАННЯМ НАВАНТАЖЕННЯ ВИКЛИКІВ МІЖ СЕРВЕРАМИ ASTERISK

2.1 Розробка мережної моделі Call-центру

Для розробки та налаштування системи моніторингу в роботі сворено мережну модель Call-центру. Мережна модель визначає загальну структуру, що моделюється: об'єкти в системі, а також їх фізичне розташування, взаємозв'язки та конфігурації. Модель може містити одну підмережу, один вузол або безліч взаємопов'язаних вузлів й підмереж. Основними структурними блоками домену мережі є підмережі, вузли та канали зв'язку. У цьому розумінні один сервер виклику становить об'єкт підмережі в домені мережі. Кластер створюється в мережному домені шляхом з'єднання кількох серверів викликів через канали зв'язку. Мережну модель Call-центру показано на рис. 2.1.

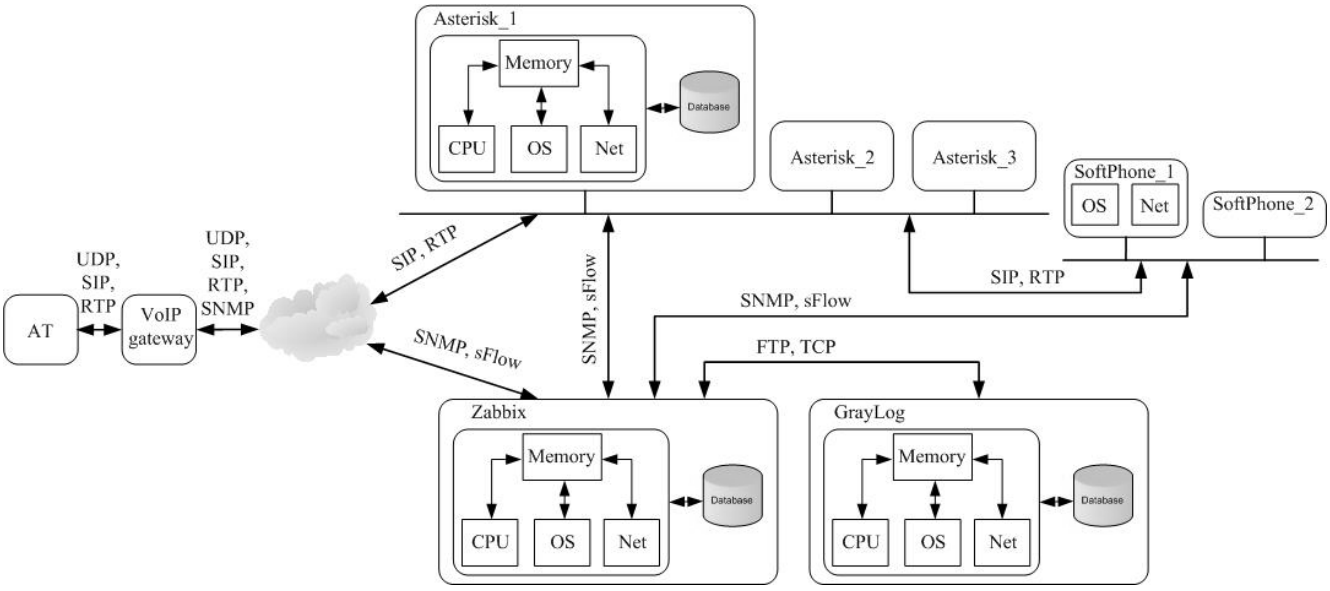

Рисунок 2.1 - Мережна модель Call-центру

Абонентський термінал являє собою пристрій, який формує виклики до АТС на виділений її зовнішній номер. На даному сегменті мережі визачено тип трафіку - це UDP, SIP, RTP, SNMP.

VoIP шлюз призначено для підключення телефонних апаратів до АТС через IP-мережу та передачі й обробки голосового трафіку.

Asterisk являє собою сервер АТС, що керує голосовим трафіком та викликами в мережі, у складі якого слід відзначити блок пам’яті, центральний процесор, мережний адаптер, операційну систему та базу даних. Характер трафіку визначається переважно протоколами SIP та RTP. В моделі показано три сервери, які об’єднано в кластер.

Zabbix являє собою сервер моніторингу, що збирає необхідні метрики з хостів мережі та записує всі події до log серверу GrayLog. Тип трафіку визначено протоколами SNMP, sFlow та SIP.

Сервер логів GrayLog з’єднано з сервером моніторингу Zabbix. Логи зберігаються в його виділеній базі даних, до якої можна звернутися у будь який момент. Основний тип трафіку: TCP, FTP.

2.2 Особливості налаштування віртуальної машини в середовищі VMWare

В роботі розроблено та налаштовано систему моніторингу з використанням open source продукту Zabbix. Зазвичай програмний продукт на основі Zabbix'a дозволяє досить просто підключати до моніторингу будь-які параметри інформаційних систем у IT-інфраструктурах, однак у роботі запропоновано його застосування для дослідження кластеру серверів викликів, які змодельовано як географічно розподілені сегменти телекомунікаційної мережі.

У роботі використано технологію віртуалізації, яка дасть змогу побудувати мережу Call-центру та провести необхідне тестування. У якості платформи для налаштування мережі використано гіпервізор VMWare ESXI 6.7 та кліент vCenter. Гіпервізор VMware ESXi – це потужний інструмент, який емулює апаратні ресурси, дозволяє безпечно виконувати машинні інструкції, ізолює та поділяє ресурси віртуальних машин. В табл. 2.1 наведено обрані параметри віртуальних машин для налаштування.

Таблиця 2.1 - Параметри віртуальних машин

Name	CPU	Memory	OS	IP-address	Hard Disk
1	2	3	4	5	6
Asterisk_1	4	6 Gb	Ubuntu 20.04	192.168.200.112	60 Gb
Asterisk_2	4	6 Gb	Ubuntu 20.04	192.168.200.104	60 Gb
Asterisk_3	4	6 Gb	Ubuntu 20.04	192.168.200.109	60 Gb

Продовження таблиці 2.1

1	2	3	4	5	6
Zabbix	6	6 Gb	Ubuntu 20.04	192.168.200.107	60 Gb
Gray_Log	4	6 Gb	Ubuntu 20.04	192.168.200.110	60 Gb
SoftPhone_1	2	4 Gb	Ubuntu 20.04	192.168.200.108	60 Gb
SoftPhone_2	2	4 Gb	Ubuntu 20.04	192.168.200.111	60 Gb
Mikrotik	1	130 Mb	RouterOS	192.168.200.254	500 Mb
SiPp	4	6 Gb	Ubuntu 20.04	192.168.100.10	60 Gb

Вікно налаштування параметрів ВМ (віртуальної машини) наведено на рис. 2.2.

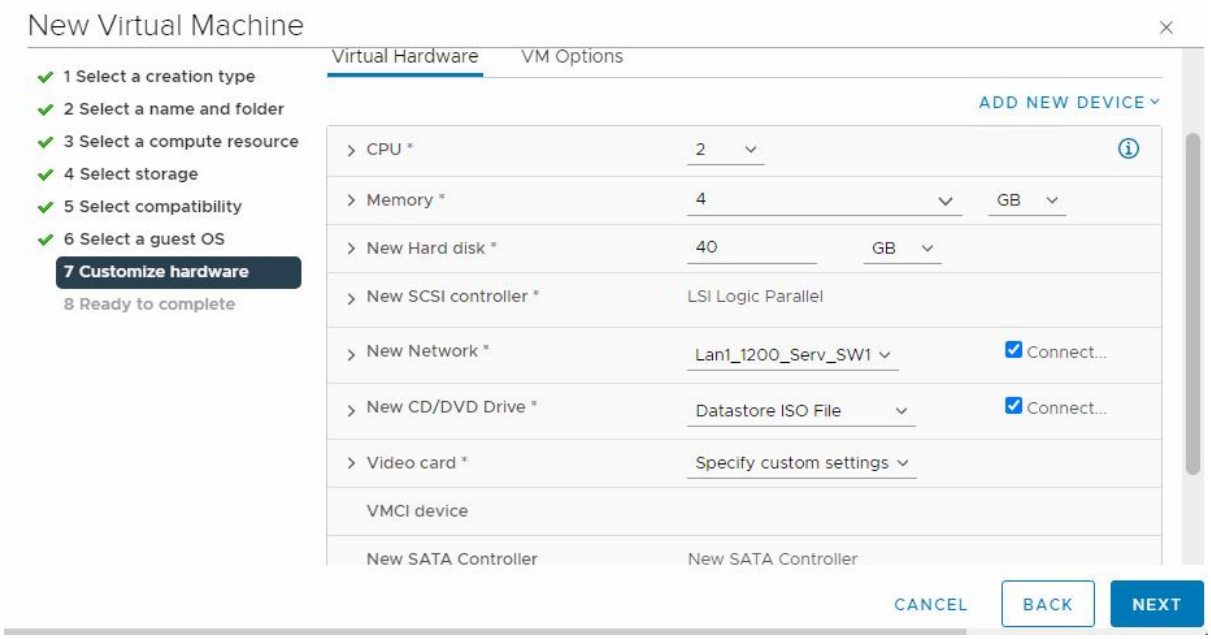

Рисунок 2.2 - Налаштування параметрів ВМ

При створенні ВМ використано наступні параметри: CPU (central processing unit), memory, місце на жорсткому диску, мережевий адаптер та ISO образ ОС (операційна система), яка буде встановлена. В роботі перевагу було надано ОС Ubuntu Server 20.04 за таких ознак: безкоштовність, досить велике коло користувачів, легкість для розгортання будь яких мережних або програмних сервісів, постійне оновлення, стабільність роботи [10, 11]. В якості ядра мережі Call-центру обрано образ RouterOS від компанії Mikrotik, основні функції якого наступніе: маршрутизація трафіку в мережі, видавання адрес хостам, виконання ролі firewall для безпеки локальної мережі.

2.3 Налаштування АТС Astersik. Створення кластеру серверів

Встановлення АТС Asterisk на сервер Ubuntu здійснювалося з готових репозиторіїв. Командою sudo apt-get install asterisk вказано системі встановити готовий пакет asterisk з офіційних джерел [12, 13]. Надалі перевіряється стан служби sudo systemctl status asterisk та проводиться конфігурація сервера.

Файли конфігурації знаходяться в каталозі /etc/asterisk. Продукт налаштовується дуже гнучко й має масу можливостей. Основними файлами конфігурації виступають sip.conf extensions.conf. Файл sip.conf відповідає за налаштування внутрішніх й зовнішніх каналів SIP в Asterisk.

Об'єкти конфігурації - піри, описуються окремих секціях, які позначаються іменами в квадратних дужках. Діє принцип успадкування, як і в більшості конфігів Asterisk: все, що задано після імені в квадратних дужках, відноситься до одного об'єкта, доки не буде оголошено наступний. Зумовлена категорія - [general], визначає глобальні налаштування драйвера SIP Asterisk, які поширюються на всі об'єкти, але можуть бути перевизначені для окремих бенкетів у їх категоріях.

Конфігурація плану набору міститься у конфігураційному файлі Asterisk - extensions.conf. Це один із найважливіших конфігураційних файлів. У ньому визначається обробка та маршрутизація вхідних та вихідних викликів. Цей файл керує поведінкою всіх з'єднань, що проходять через АТС. Зміст файлу extensions.conf розбито на секції, в яких можуть бути або визначені статичні налаштування та визначення або команди плану набору, що виконуються, в цьому випадку вони називаються контекстами. Секції, призначені для статичних налаштувань, називаються general і globals, а імена контекстів визначаються системним адміністратором системи.

В роботі було прийнято рішення обмежитись лише двома віртуальними телефонами для перевірки викликів в мережу. Тому при конфігурації файлу sip.conf, додано лише два записи для номерів 201 та 202. Приклад запису в файлі extensions.conf показано на рис. 2.3.

```
[general]
static=yes
writeprotect=yes
[gloabal]
[default]
exten => 201,1,Dial(SIP/201)
exten => 202,1,Dial(SIP/202)

exten => _[34]XX,1,Dial(SIP/asterisk2/${EXTEN},15,Tt)
exten => _[34]XX,1,Dial(SIP/asterisk3/${EXTEN},15,Tt)

[asterisk2-in]
exten => _XXX,1,Dial(SIP/${EXTEN},15,Tt
[asterisk3-in]
exten => _XXX,1,Dial(SIP/${EXTEN},15,Tt
```

Рисунок 2.3 - План набору в файлі extensions.conf

В файлі extensions.conf. прописано план набору для телефонів 201 та 202.

На наступному кроці налаштовано sip транки для об’єднання серверів Asterisk в кластер. Для цього на кожному з серверів введено новий пір, який буде використовувати сервер для підключення до наступного. Тобто налаштовується реєстрація серверів меж собою, але без підтвердження. Усі необхідні налаштування проведено в файлах sip.conf та exstensions.conf. На рис. 2.4 показано процес перевірки реєстрації серверів між собою на прикладі Astersik_1.

```
aster1*CLI> sip show peers
Name/username             Host                                    Dyn Forcerport Comedia    ACL Port     Status
ion
201/201                   192.168.200.113                          D  Auto (No)  No             57085    Unmonitored

202/202                   192.168.200.100                          D  Auto (No)  No             50981    Unmonitored

asterisk2                 192.168.200.104                             Auto (No)  No             5060     OK (1 ms)

asterisk3                 192.168.200.109                             Auto (No)  No             5060     OK (1 ms)

4 sip peers [Monitored: 2 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Рисунок 2.4 - Інформація про реєстрацію в системі на прикладі Astersik_1

Приклад перевірки з’єднання наведено на рис. 2.5.

```
  == Using SIP RTP CoS mark 5
     > 0x7f324c04f230 -- Strict RTP learning after remote address set to: 192.168.200.113:42694
    -- Executing [202@default:1] Dial("SIP/201-00000008", "SIP/202") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/202
    -- SIP/202-00000009 is ringing
     > 0x7f3280017590 -- Strict RTP learning after remote address set to: 10.100.0.128:4004
    -- SIP/202-00000009 answered SIP/201-00000008
    -- Channel SIP/202-00000009 joined 'simple_bridge' basic-bridge <f35862f0-0456-4f7a-a910-e47f1232a130>
    -- Channel SIP/201-00000008 joined 'simple_bridge' basic-bridge <f35862f0-0456-4f7a-a910-e47f1232a130>
     > Bridge f35862f0-0456-4f7a-a910-e47f1232a130: switching from simple_bridge technology to native_rtp
     > Remotely bridged 'SIP/201-00000008' and 'SIP/202-00000009' - media will flow directly between them
     > 0x7f3280017590 -- Strict RTP learning after remote address set to: 10.100.0.128:4004
     > 0x7f324c04f230 -- Strict RTP switching to RTP target address 192.168.200.113:42694 as source
    -- Channel SIP/202-00000009 left 'native_rtp' basic-bridge <f35862f0-0456-4f7a-a910-e47f1232a130>
    -- Channel SIP/201-00000008 left 'native_rtp' basic-bridge <f35862f0-0456-4f7a-a910-e47f1232a130>
  == Spawn extension (default, 202, 1) exited non-zero on 'SIP/201-00000008'
    -- Remote UNIX connection
Asterisk Queue Logger restarted
    -- Remote UNIX connection disconnected
```

Рисунок 2.5 - Процес ініціювання з’єднання 202 номера з 201

2.4 Налаштування SoftPhone

В якості IP телефонів в роботі було використано програмне забезпечення Zoiper5. На відміну від більшості тестових стендів, на ВМ встановлено графічну оболонку XFCE для взаємодії з програмою.

В Zoiper5 варто запинитись на пункті з реєстрацією. В програмі в налаштуванні вводяться наступні пункти: account type – тип протоколу, що використовується при підключенні, user@host – ім'я користувача та sip сервер, password – пароль, domain / outbound proxy – ip адреса/ доменне ім'я сервера або проксі. Приклад реєстрації в АТС в програмі Zoiper наведено на рис. 2.6.

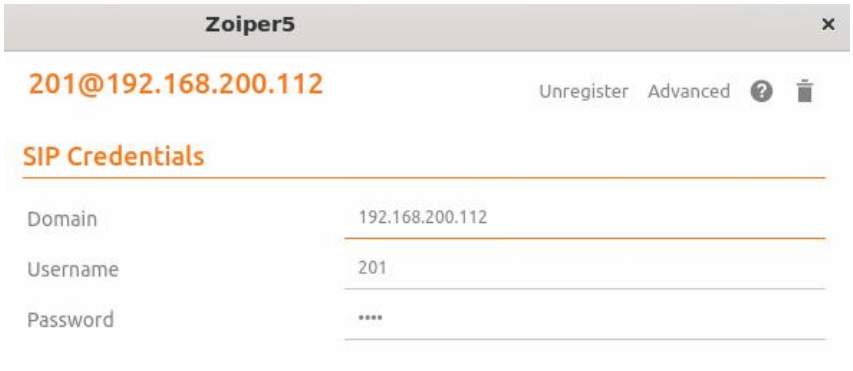

Рисунок 2.6 - Процес реєстрації в АТС в програмі Zoiper

2.5 Створення Zabbix серверу та додавання host моніторингу

Сервер Ubuntu 20.04 створено відповідно до пункту 2.1. Початкове налаштування сервера для Ubuntu 20.04 включає користувача без привілеїв root з привілеями sudo та налаштований брандмауер ufw. Основна задача сервера Zabbix - це моніторинг всієї мережі Call-центру, зняття необхідних метрик (стан каналів, стан sip транків, пропускна здатність каналів, кількість втрачених пакетів, навантаження ВМ). Доменне ім'я серверу в роботі не виділяється. Підключення виконується за його IP адресою та портом.

На початку встановлення Zabbix з дистрибутива встановлюється та налаштовується в середовищі Ubuntu веб сервер apache, база данних MySql та PHP (personal home page), що необхідні для обробки коду та створення динамічного контенту для веб-сервера [14]. Всі керування процесом моніторингу виконуються саме в графічному середовищі з браузеру.

Наступним етапом розгорнута платформа за допомогою репозиторіїв. Zabbix має чотири основні інструменти, за допомогою яких можна моніторити певне робоче середовище й збирати про нього повний пакет даних для оптимізації роботи [15].

Сервер - це ядро, що зберігає в собі всі ці системи, включаючи статистичні, оперативні та конфігурацію. Дистанційно керує мережними сервісами, повідомляє адміністратору про існуючі проблеми з обладнанням, яке перебуває під наглядом.

Проксі - це сервіс, що збирає дані про доступність та продуктивність пристроїв, який працює від імені сервера. Усі зібрані дані зберігаються у буфер і завантажуються на сервер. Завдяки цьому можна зменшити навантаження на процесор та жорсткий диск. Для роботи проксі Zabbix окремо потрібна база даних.

Агент - програма (демон), яка активно моніторить та збирає статистику роботи локальних ресурсів (накопичувачі, оперативна пам'ять, процесор та ін.) та додатків.

Веб-інтерфейс є частиною сервера системи й вимагає для роботи веб-сервер. На рис. 2.7 представлено вікно графічного інтерфейсу Zabbix при встановленні системи.

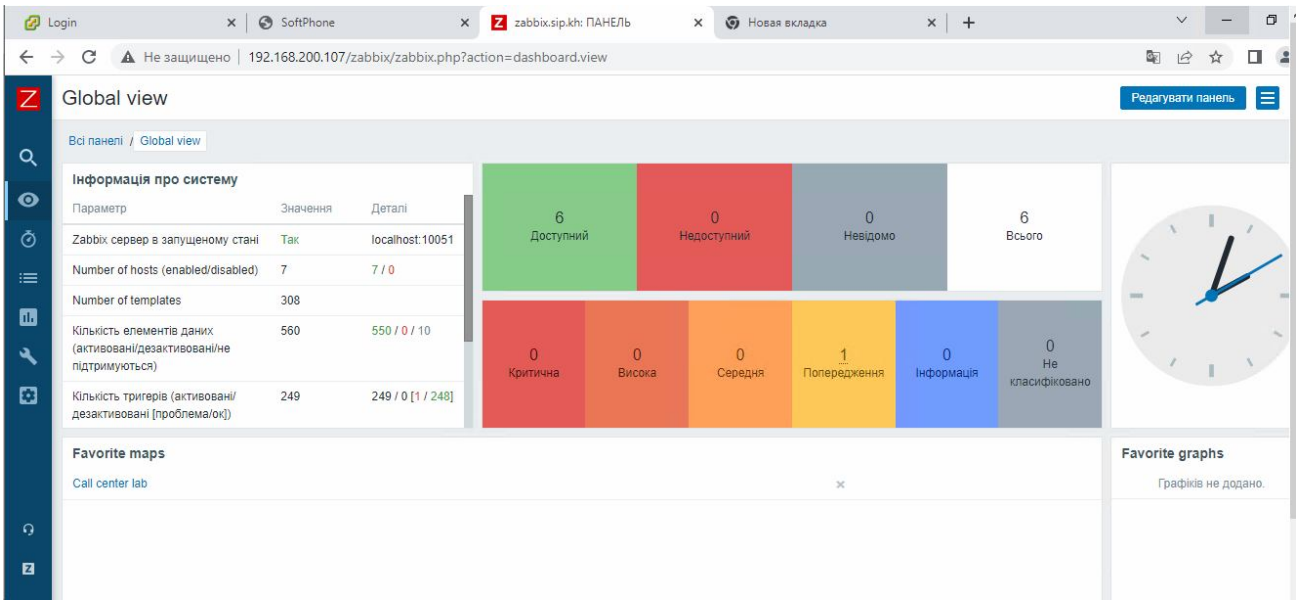

Рисунок 2.7 - Початкова сторінка Zabbix

Для того, щоб мати змогу проводити моніторинг обладнання в мережі, його потрібно додати з відповідними параметрами. На рис. 2.8 показано процес додавання нового host до системи.

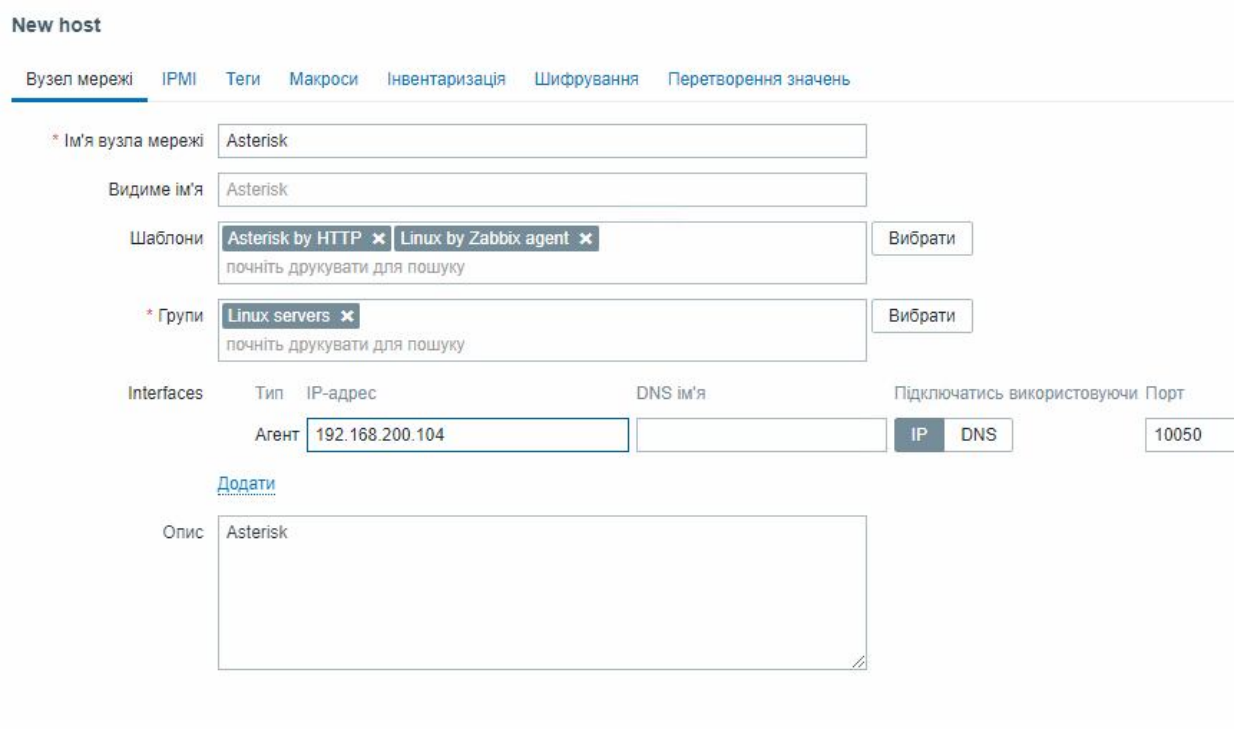

Рисунок 2.8 - Процес додавання host моніторингу

В формі вказується ім’я host, шаблони, за якими буде моніторитись host, група для структуризації, інтерфейс або інтерфейси host та опис.

Тригери в Zabbix - це логічні висловлювання, які відображають стан системи. Тригер може приймати такі значення: "Проблема", "Ок", "Невідомо". Висловлювання, що використовуються в тригерах, є дуже гнучкими. Адміністратор може використовувати їх для створення складних логічних тестів з огляду на статистику з моніторингу [16].

Функції тригерів дозволяють посилатися на зібрані значення, поточний час та інші фактори. Стан (вираз) тригера перераховується щоразу, коли Zabbix сервер отримує нове значення даних, якщо це значення даних є частиною висловлювання. Якщо у виразі тригера використовуються функції, що відносяться до часу, такий тригер перераховується кожні 30 секунд. Функції, що стосуються часу: nodata(), date(), dayofmonth(), dayofweek(), time(), now() повертають лише числові значення. Порівняння рядків, наприклад, не підтримується. Рядкові аргументи повинні бути укладені в подвійні лапки. Інакше вони можуть бути

неправильно інтерпретовані. Важливість тригера визначає, наскільки тригер важливий. Zabbix підтримує такі важливості тригерів: невідома важливість; в інформаційних цілях; попереджувальний; середня проблема; сталося щось важливе; надзвичайний [17].

На рис. 2.9 зображено приклад трігера 'www.zabbix.com:system.cpu.load[all,avg1]', що передає коротке ім'я параметра для спостереження.

```
{www.zabbix.com:system.cpu.load[all,avg1].last(0)}>5
```

Рисунок 2.9 - Приклад трігера

Цей рядок вказує, що контролюється сервер 'www.zabbix.com' та ключ 'system.cpu.load[all, avg1]'. Використовуючи функцію 'last()' є посилання на останнє значення. І нарешті '>5' означає, що тригер буде визначений як "Проблема" щоразу, коли останнє значення завантаження процесора на сервері www.zabbix.com буде перевищувати 5.

На рис. 2.10 показано приклад тригерів, які використовуються для Asterisk, їх можна вибрати з шаблону при налаштуванні моніторингу host.

Важность	Имя ▲	Выражение
Высокая	Asterisk down on {HOST.NAME}	{Asterisk:asterisk.asterisk_status.last()}=0
Информация	Asterisk restarted on {HOST.NAME}	{Asterisk:asterisk.uptime.last()}<300
Средняя	Fail2ban down on {HOST.NAME}	{Asterisk:asterisk.fail2ban_status.last()}=0
Средняя	Fail2ban inactive on {HOST.NAME}	{Asterisk:asterisk.fail2ban_chain.last()}=0
Средняя	Trunk not registered on {HOST.NAME}	Проблема: {Asterisk:asterisk.trunk.count(#2,"All trunks are online","like")}=0 Восстановление: {Asterisk:asterisk.trunk.count(#2,"All trunks are online","like")}=2

Рисунок 2.10 - Тригери для серверів Asterisk

У шаблоні присутні шість елементів даних, які визначаються в агенті, п’ять тригерів та один графік. Опис кожного з тригерів показано в Додатку А.

Віджети, графіки та карти мереж в Zabbix потрібні для візуалізації даних. Це насамперед полегшує сприйняття інформації адміністратором системи. Після налаштувань для візуалізації даних в Zabbix отримано графіки та карту мережі. Карта мережі являє собою змодельовану структуру мережі Call-центру.

На рис. 2.11 показан графік з трафіком серверу для Asterisk_1.

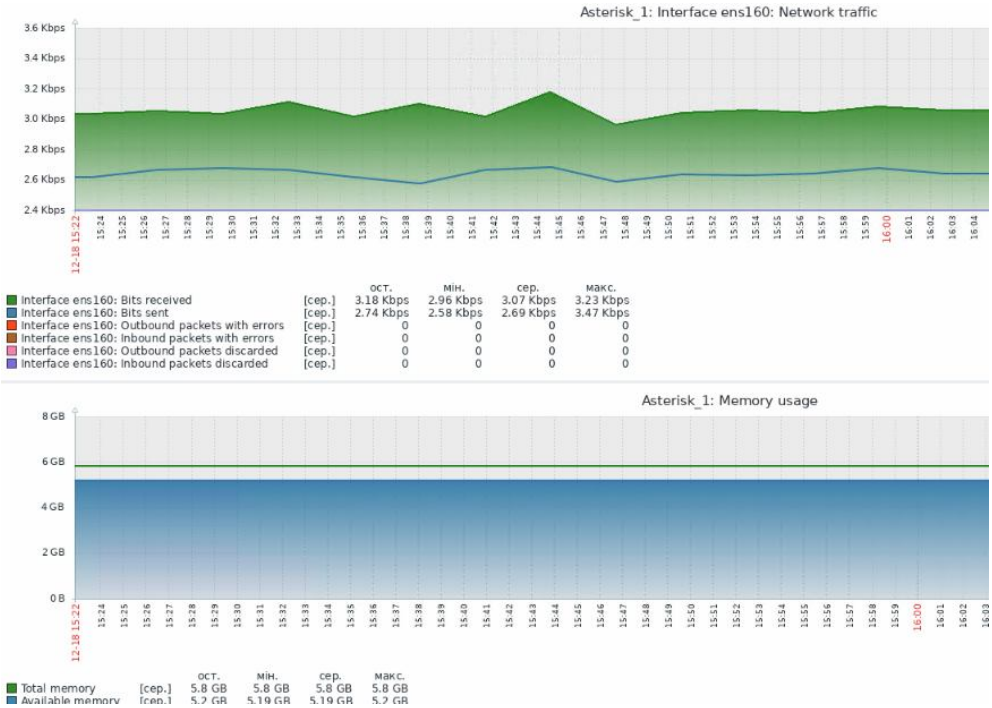

Рисунок 2.11 - Графіки пропускної здатності інтерфейсу ens160 та пам'яті сервера Astersik_1

На рис. 2.12 Показано приклад створення графіку викликів для Asterisk_1.

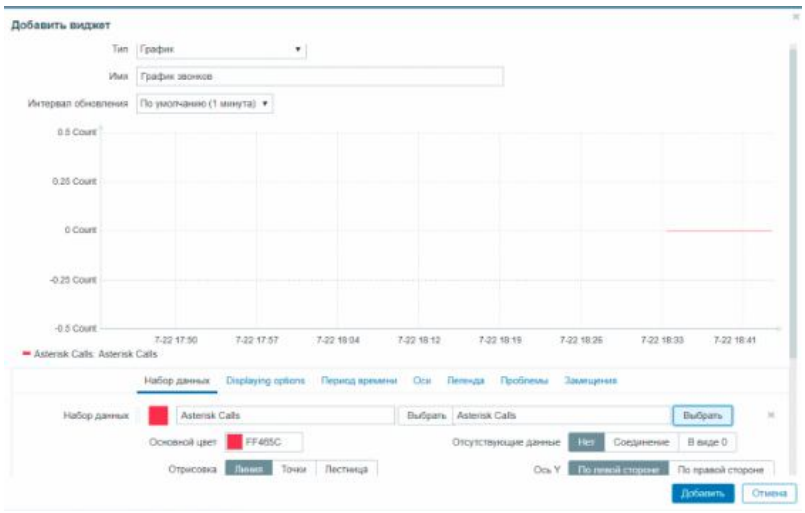

Рисунок 2.12 - Налаштування графіку в Zabbix на прикладі графіку викликів

На рис. 2.13 показано топологію мережі Call-центру (карту мережі Call-центру), яку отримано в результаті налаштувань. Для прикладу телефон з адресою 192.168.200.113 підсвічено жовтим кольором (в його системі заздалегідь був налаштований інший час). Видно, що даний тригер відображається на мапі як попереджувальний та відображає коротку інформацію адміністратору.

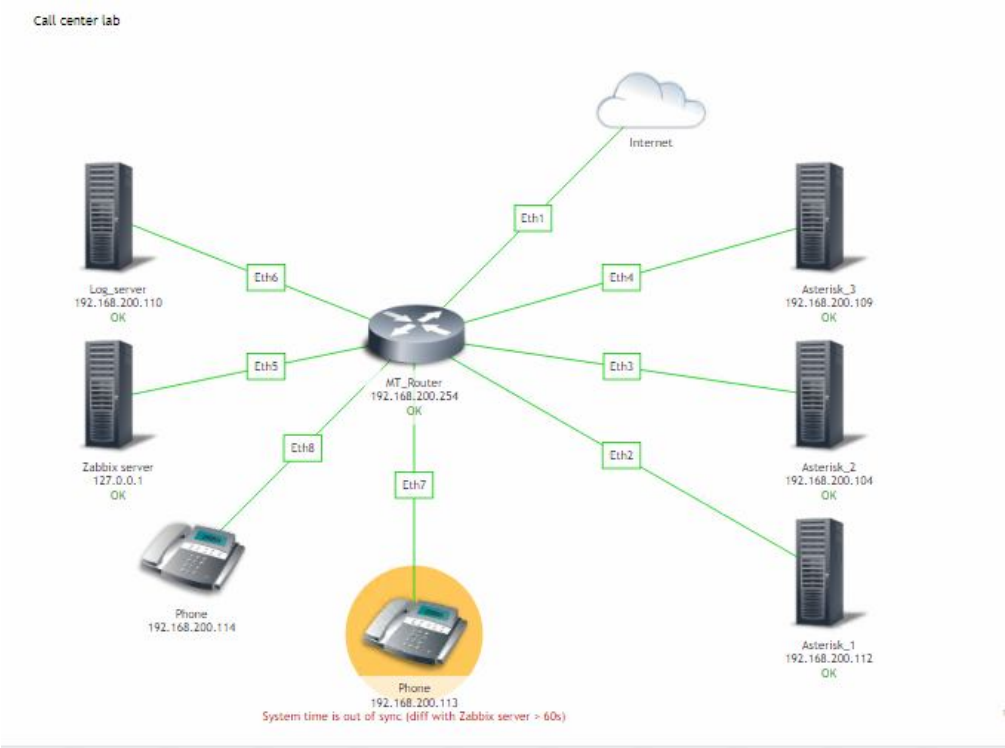

Рисунок 2.13 - Карта мережі Call-центру

2.6 Аналіз та керування балансуванням системою Zabbix

В роботі показано процес тестування навантаження викликами на три сервера Asterisk та реалізація можливостей Zabbix як інструменту балансування навантаження. При цьому розрахована кількість викликів, з якими здатний працювати Call-центр.

Слід зазначити, що Zabbix створено саме як інструмент моніторингу, який збирає необхідні метрики в системі та надає їх адміністратору системи. Проте, його може бути певним чином сконфігуровано для того, щоб реагувати на певні прояви системи. Робиться це за допомогою скриптів та трігерів.

В мережі Call-центру ВМ Asterisk_1 виступає як в ролі АТС, так і в ролі сервера балансування навантаження. В якості програмного забезпечення, що відповідає за балансування навантаження, на сервері встановлено та налаштовано модуль Kamailio. За замовчуванням його вимкнено. В роботі прийнято, що Zabbix, перевіряючи стан навантаженості процесору, увімкне модуль Kamailio, якщо його значення буде більше або дорівнювати 90 %.

Тестування навантаження відбувається за допомогою утиліти SIPp. SIPp - потужна утиліта для створення навантаження на SIP обладнання. Зазвичай SIPp використовується для перевірки відмови системи IP-телефонії, виявлення

максимально допустимого навантаження або DDoS-атак (distributed denial-of-service). Сценарій сесії в SIPp описується в XML (extensible markup language) файлі. Можливо скористатися одним із безлічі сценаріїв, що розповсюджуються в комплекті з SIPp або створити свій сценарій сесії в SIPp.

Для того, щоб Asterisk приймав виклики від SIPp, створено в SIP.conf спеціальний SIP-peer з ім'ям sipp (рис. 2.14).

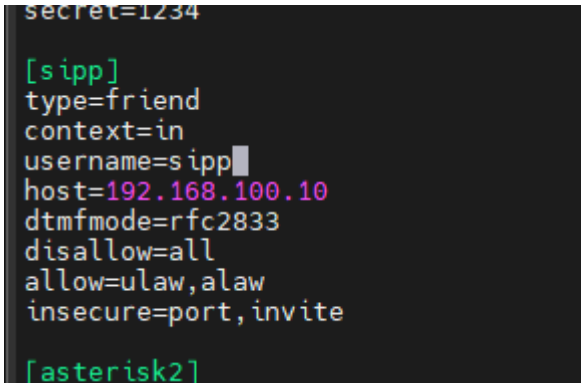

Рисунок 2.14 - Налаштування SIP-peer для SIPp

Важливим моментом є наявність кодека ulaw у списку дозволених, тому що саме його анонсує SIPp. Якщо 711u не буде в списку дозволених кодеків, Asterisk відхиляє виклик від SIPp. Другим важливим моментом є рядок insecure=port,invite. Цей рядок змушує Asterisk авторизувати SIPp не за паролем, а за IP адресою, яку вказано в полі host. Крім запису в SIP.conf, можна створити спеціальний контекст extensions.conf для обробки тестових викликів від SIPp. На рис. 2.15 показано процес додавання запису в файл extensions.conf.

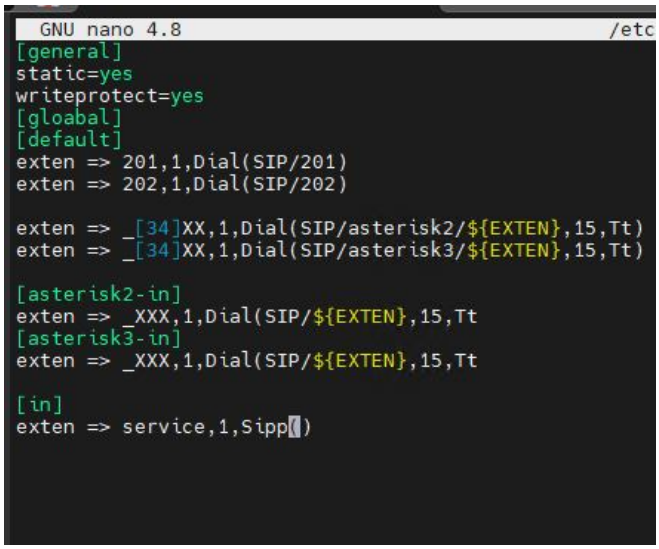

Рисунок 2.15 - Додавання запису в extensions.conf для налаштування SIPp

В роботі створено власний сценарій сесії в SIPp, приклад налаштування параметрів якого наведено на рис. 2.16.

```
sipp@phone:/etc$ cd /etc/sipp-3.3
sipp@phone:/etc/sipp-3.3$ sudo sipp 192.168.200.112 -s 201 -i 192.168.100.10 -d 2h -l 60 -aa -mi 10.10.10.2 -rtp_echo -nd -r
10
```

Рисунок 2.16 - Команда для старту відправки викликів SIPp з заданими параметрами

Параметри даної команди наведено в Додатку Б.

На рис. 2.17 показано фрагмент роботи утиліти SIPp при заданих параметрах навантаження.

```
1 calls (limit 2)                    Peak was 2 calls, after 30 s
1 Running, 4 Paused, 3 Woken up
0 dead call msg (discarded)          390 out-of-call msg (discarded)
3 open sockets
1309 Total RTP pckts sent            0.000 last period RTP rate (kB/s)

                          Messages  Retrans   Timeout   Unexpected-Msg
    INVITE ---------->         7         0         0
       100 <----------         7         0         0         0
       180 <----------         7         0         0         0
       200 <----------  E-RTD1 6         0         0         0

       ACK ---------->         6         0
           [ NOP ]
     Pause [   8000ms]         6                             2
           [ NOP ]
     Pause [   1000ms]         4                             0
       BYE ---------->         4         0         0
       200 <----------         4         0         0         0

------ [+|-|*|/]: Adjust rate ---- [q]: Soft exit ---- [p]: Pause traffic -----
```

Рисунок 2.17 – Фрагмент роботи утиліти SIPp

В процесі навантаження кількість викликів поступово збільшувалась на 50. Виявилось, що один сервер Asterisk з поточними його параметрами здатний обробити максимум 915 одночасних викликів. Результати тестування показано на графіку (рис. 2.18).

Виходячи з результатов, отриманих на графіку, для сервера Asterisk було встановлено порогове значення завантаженності CPU в 90 % (йому відповідає 850 викликів). Саме при ньому за для уникнення повного перевантаження серверу потрібно запускати в роботу додаткові сервери.

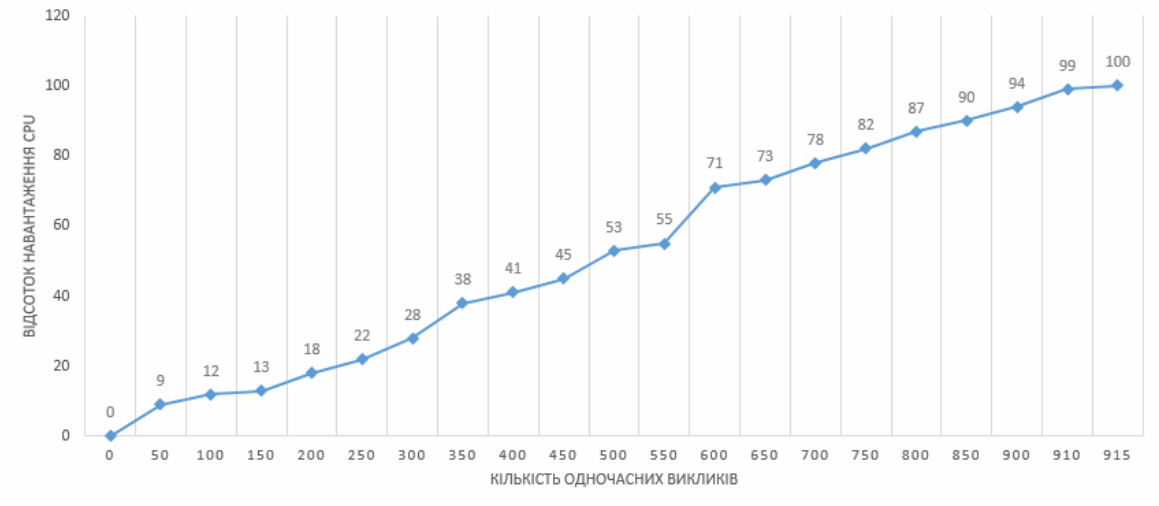

Рисунок 2.18 - Графік залежності навантаження CPU від кількості викликів

В роботі за допомогою системи моніторингу запускається процес балансування навантаження на кластер SIP серверів, виходячи з метрик стану CPU та стану SIP транку. Для того, щоб Zabbix мав змогу перевіряти стан CPU та стан транку, розроблено необхідний скріпт та додано його в існуючі параметри моніторингу host.

В мережі Call-центру ВМ Asterisk_1 виступає як в ролі АТС, так і в ролі сервера балансування навантаження. В якості програмного забезпечення, яке відповідає за балансування, на сервері встановлено та налаштовано модуль Kamailio. За замовчуванням його вимкнено.

Для реалізації заданих умов перевірки використано можливості скриптів та тригерів в Zabbix. При виявленні заданого відсотка завантаженності CPU (в даному випадку 90 %) налаштовано скрипт, за яким Zabbix відправляє команду до серверу Asterisk на увімкнення даного модуля.

На рис. 2.19 показано процес створення тригеру, котрий реагує на завантаження CPU host Asterisk.

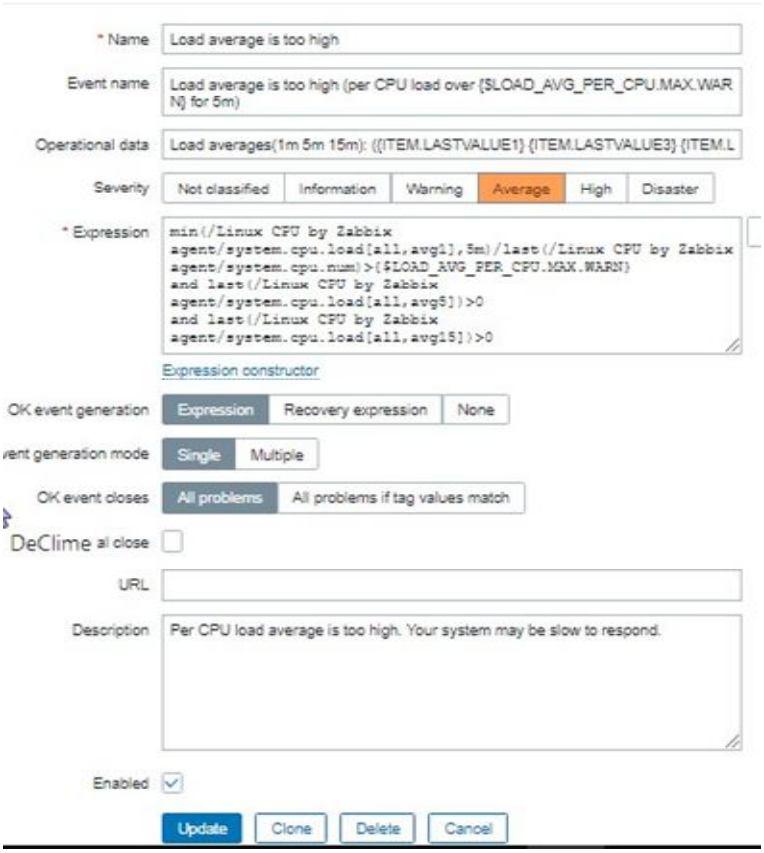

Рисунок 2.19 - Процес створення тригеру навантаження CPU

На рис. 2.20 показано налаштування дії на розроблений тригер.

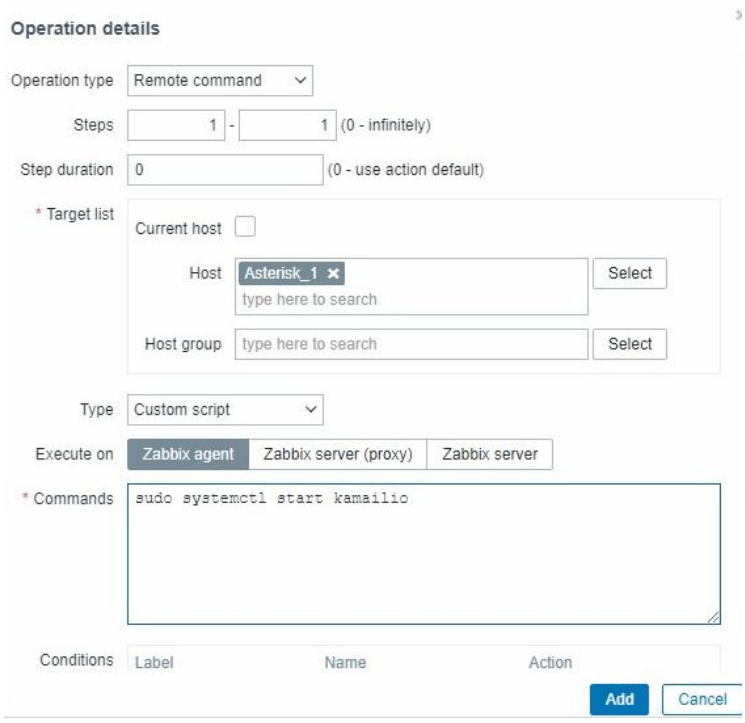

Рисунок 2.20 - Налаштування дії на відправку команди для увімкнення утиліти Kamailio

Таким чином, в мережі Call-центру налаштовано можливості системи моніторинга з використанням Zabbix для балансування навантаження в кластері серверів Astersik. Було виявлено, що кластер серверів Asterisk здатний обслуговувати 2550 викликів одночасно.

3 КЛАСТЕРИЗАЦІЯ CALL SERVER. ДОСЛІДЖЕННЯ РОБОТИ ОДИНОЧНОГО CS У КЛАСТЕРІ

3.1 Аналіз методів розподілу навантаження у кластері серверів

Необхідними умовами для ефективного розподілу навантаження або балансування навантаження є адаптація методів розподілу навантаження до області кластерів серверів викликів та дослідження політик переадресації.

Методи балансування навантаження поділяються на дві основні категорії: статичні та динамічні.

У статичних методах використовуються зумовлені формули чи числа без використання будь-якої інформації про стан системи під час виконання. Статичні алгоритми прості у реалізації. Прикладами статичного розподілу навантаження є: розподіл навантаження одним брокером за круговою схемою; випадковий розподіл навантаження Random, кругове балансування навантаження Round Robin, виважений розподіл навантаження Weighted Round Robin тощо. Статичний розподіл навантаження вважається ефективним за наявності лише одного централізованого брокера або диспетчера завдань та завдань однакового розміру (з рівним часом обробки). Цей підхід вважається дуже корисним для розподіленого балансування навантаження [18, 20].

Методи динамічного балансування навантаження використовують оперативну інформацію про стан системи прийняття рішень. Прийняття рішень можливо використовувати на різних рівнях: на рівні завантаження ЦП (центрального процесора), доступної пам'яті тощо [20]. Методи динамічного балансування навантаження називають методами адаптивного розподілу навантаження [18,19]. У методах адаптивного розподілу навантаження продуктивність системи може бути використана для коригування розподілу навантаження під час роботи.

У роботі проведено аналіз балансування навантаження з використанням різних алгоритмів та стратегій. У літературі [21] запропоновано використовувати двокаскадний комутатор з балансуванням навантаження для масштабування до швидкості оптоволокна. Такий підхід вважається більш ефективним, ніж інші підходи, такі як i-SLIP, що називається алгоритмом вирішення конфліктів. Його засновано на ітеративному циклічному зіставленні з ковзанням. У літературі [22]

паоказно, що для трафіку з багатопротокольною комутацією за мітками MPLS (multiprotocol label switching) алгоритми динамічного розподілу навантаження краще за продуктивністю, ніж алгоритм найкоротшого шляху. Зазначено, що алгоритми динамічного балансування навантаження ефективні як з легких, так і з важких умов навантаження.

Алгоритм перенаправлення запитів для кластерів веб-серверів досліджено в [23]. Проведено класифікацію кожного нового запиту на основі очікуваного впливу на ресурси сервера. При цьому кожному запиту надається вага, яка використовується для розрахунку навантаження на сервер. Щоб скоротити час збору інформації про навантаження, вибираються лише два сервери з пулів серверів, і новий запит перенаправляється на найменш завантажений сервер.

У [24] запропоновано використовувати стратегії розподілу навантаження, що засновано не тільки на ресурсах ЦП, але й із урахуванням ресурсів пам'яті. Метою запропонованих стратегій розподілу навантаження є як мінімізація часу простою ЦП, так і кількості відмов сторінок в гетерогенних розподілених системах.

У [25] використано мобільні агенти для забезпечення розподілу навантаження у глобальному мережному середовищі, такому як мережа Інтернет. Мобільні агенти діють як координатори від імені серверів і представляють себе на віддалених майданчиках для координації дій щодо переміщення робочих місць.

3.2 Політики розподілу навантаження

Політики розподілу навантаження складаються із чотирьох компонентів [26]: політики перенесення, політики переадресації, політики міграції та інформаційної політики.

Політика перенесення визначає, коли переносити навантаження до іншого вузла. Зазвичай, диспетчер навантаження або брокер приймає це рішення на основі інформації про навантаження локального вузла. У рамках політики перенесення відомо прості та ефективні підходи, які засновано на фіксованому порозі та адаптивному порозі [27, 28].

Перший підхід проводить оцінку поточної інформації про навантаження локального вузла з певним фіксованим порогом. Якщо поточне навантаження перевищує фіксований поріг, локальний вузол намагається передати частину свого навантаження іншому вузлу.

Динамічна чи адаптивна політика переносу з урахуванням порога спочатку визначає поріг з урахуванням інформації про навантаження всіх вузлів у системі. Якщо поточне навантаження вузла вище цього "оцінного" порога, він починає передавати навантаження на слабонавантажений вузол.

Помічено, що проста політика з фіксованим порогом дає кращі результати ніж адаптивна політика на основі порога [28]. Для розподілу навантаження в кластері CS (Call Server) у роботі розглядається політика з урахуванням фіксованого порога. Сервер обробки викликів працює на рівні проєктної потужності.

Політика міграції визначає, звідки чи куди має бути перенесене завдання. Політики міграції класифікуються: ініційовані відправником, ініційовані одержувачем й симетрично ініційовані [29, 30]. У політиках ініційованих відправником потенційні відправники завдань шукають одержувачів, на які може бути перенесено навантаження. У політиці, що ініціюється одержувачем [30], потенційні одержувачі завдань шукають відправників. У симетрично ініційованій політиці і відправники, і одержувачі шукають додаткові вузли. У роботі прийнято політику, ініційовану відправником, враховуючи природу політик переадресації. Ключовим аспектом політики міграції, що ініціюється відправником, є критерії, які використовуються для вибору одержувача.

Першим кроком є пошук потенційних одержувачів у кластері Call Server. Пороговий підхід найпоширеніший для вибору потенційного приймача [31]. Якщо навантаження на вузол менше певного порога, цей вузол вважається потенційним одержувачем.

Наступним кроком є вибір одного чи кількох одержувачів зі списку потенційних одержувачів. Важливим моментом тут є те, що кілька відправників можуть вибрати одного одержувача та почати передачу завдань одночасно. Це спричинить перевантаження приймача. Цю проблему можна звести до мінімуму, якщо перевірити приймач перед передачею на нього навантаження.

Існує кілька підходів для роботи політики міграції:

1) Адаптивна політика міграції на основі передбачення [32]. У цьому випадку всі вузли з'єднано у вигляді кільця, і відправник шукатиме зі списку потенційних одержувачів той вузол, який попереду його в кільці.

2) Вибір приймачів випадково зі списку потенційних приймачів, а потім вибір найменш завантаженого одержувача зі списку. Політика вибору визначає, яке завдання має бути передане для віддаленого виконання.

У роботі використано стратегію, засновану на зваженому випадковому виборі одержувача зі списку приймачів. І тому визначаються доступні можливості приймача. Доступність приймача – це різниця між порогом політики міграції та індексом завантаження приймача CS. Такий алгоритм отримує список одержувачів зі своїми доступними можливостями з урахуванням порога політики міграції. Одержувач у списку вибирається випадково.

Можлива як попереджувальна, так і не попереджувальна політика [33]. У цьому випадку вибір завдань для передачі базується на критеріях, які використовують різні характеристики: обсяг пам'яті, частоту доступу введення-виведення та час виконання [34, 35].

Політика переадресації в кластерах Call Server повністю зосереджена на визначенні того, яка частина нових викликів має оброблятися локально, а яка частина має передаватися на віддалений Call Server. Ця політика використовується для визначення кількості нових вхідних викликів, що очікують обробки на сервері обробки викликів, і які повинні бути перенаправлені на інший сервер обробки викликів для розподілу навантаження всередині кластера.

Відомі адаптивні методи, що засновано на інтелектуальному аналізі даних, визначення кількості завдань, які необхідно перенести на різні вузли в розподіленій системі [36]. Це досягається використанням центрального диспетчера завдань для переадресації нових завдань на сервери та балансувальника задач для перенесення завдань між різними серверами, тобто балансування навантаження. У такій системі центральний диспетчер завдань є єдиною точкою відмови та може стати вузьким місцем у продуктивності. Проте використання балансувальника навантаження передбачає кілька алгоритмів до роботи системи: алгоритми множинної передачі завдань іншим компонентам у системі з навантаженням, тобто вибір кількох завдань ґрунтується на їх залежності від інших вузлів у системі; використання алгоритмів перенаправлення трафіку веб-сервера [37].

Інформаційна політика стосується того, як і коли обмінюватись інформацією про навантаження між різними вузлами системи [38]. Існує три різні типи інформаційної політики:

1) Періодична інформаційна політика, за якої або кожний вузол передає свою інформацію про навантаження іншим вузлам через фіксований період часу, або кожний вузол періодично перевіряє інші вузли в системі для збору інформації про навантаження.

2) Інформаційна політика, керована попитом, у якій інформація про завантаження витягується з інших вузлів лише тоді, коли це потрібно. Якщо вузол перевантажений, він перевіряє інші вузли у системі, щоб знайти найкращого одержувача для передачі навантаження. Одним із недоліків цього підходу є додаткова затримка, тому що відправник повинен збирати інформацію про завантаження інших вузлів під час виконання.

3) Інформаційна політика на основі зміни стану, в якому інформація про навантаження передається іншим вузлам лише у тому випадку, якщо навантаження на вузол змінюється на певний ступінь. Кожний вузол періодично оцінює стан свого навантаження, але передає інформацію про навантаження лише у тому разі, якщо зміна стану навантаження перевищує фіксований поріг.

3.3 Концепція кластеризації Call Server

Сервер викликів CS - це елемент, що складається з механізму керування викликами, який виконує їх обробку. Кластер серверів викликів є групою з декількох окремих серверів викликів, що охоплюють певну географічну область. Кластеризація Call Server забезпечує переваги як масштабованості, так і надійності.

Балансування навантаження є важливим аспектом кластерів Call Server. Один із серверів обробки викликів у кластері може бути перевантажений, у той час як інші сервери обробки викликів можуть працювати нижче за призначений рівень навантаження. У разі такої незбалансованої ситуації можна розподілити додатковий трафік перевантаженого Call Server різним Call Server у кластері. Такий рівень розподілу навантаження не дозволяє одному серверу викликів стати вузьким місцем у продуктивності.

У роботі основну увагу приділено розподілу навантаження у кластерах Call Server для досягнення високої продуктивності системи. Зазвичай новий виклик обробляється сервером викликів, який містить дані абонента, пов'язані з ініціатором виклика. Але у разі перевантаження деяка кількість викликів, що очікують обслуговування, переводиться на слабозавантажений сервер обробки викликів. Такий механізм вимагає згладженої роботи системи балансування навантаження: прийняття рішення про те, коли розпочинати передачу навантаження з вузла, що контролюється політикою перенесення; спільного використання інформації про навантаження між вузлами, що регулюється

інформаційною політикою; визначення того, звідки чи в яке місце має бути перенесене завдання, що потребує використання політики міграції.

Аналіз компонентів системи розподілу навантаження показав, що існує багато проблем, що стосуються їх одночасної роботи [18, 19]. Виявлено, що розподіл навантаження здійснюється шляхом передачі одного завдання за один раз. У разі великої кількості завдань, що чекають на перевантаженому вузлі, перенесення одного завдання за один раз є дорогим і може не дозволити своєчасно зняти навантаження з перевантаженого вузла. Ця здатність виконувати своєчасну передачу навантаження є особливо важливою для вирішення завдань системи в реальному масштабі часу. Тому в таких ситуаціях дуже важливо передавати кілька завдань одночасно, щоби прискорити розподіл навантаження.

Для ефективного розподілу навантаження потрібні стратегії, які зосереджені на дослідженні різних методів розрахунку частини нових вхідних викликів, які мають бути перенаправлені на інший сервер викликів для досягнення високої продуктивності. Політика, яка використовується для визначення кількості викликів, які повинні бути перенаправлені з перевантаженого вузла на інший слабозавантажений вузол, називається політикою переадресації телекомунікаційної системи.

3.4 Кластер серверів викликів

Концепцію кластерів Call Server прийнято розглядати у зв'язку з поняттям "телекомунікаційна система". Тут важливу роль відіграє телекомунікаційний комутатор – це система, яка забезпечує з'єднання між двома або більше абонентами для обміну голосом та даними. Сервер обробки викликів та комутаційна мережа є двома важливими компонентами телекомунікаційного комутатора.

CS виконує обробку викликів, а також різні дії, що пов'язані з білінгом, в той час як мережа комутації відповідає за забезпечення шляху для з'єднання абонентів. У кластері серверів викликів група серверів викликів підключається до мережі комутації, яка забезпечує структуру комутації, що необхідна для підключення абонентів. Сервери у кластері взаємодіють один з одним через виділену мережу з'єднань.

Кластеризація серверів викликів може здійснюватись на різних рівнях. Вона може включати частини CS, змонтовані на одному кадрі, кілька кадрів Call Server

на одному сегменті, а також географічно розподілені сегменти, кожен з яких містить один або кілька Call Server.

Для зв'язку між абонентами, що належать різним кластерам CS, використовуються магістралі, що з'єднують між собою кластери CS. Коли абонент знімає слухавку, функція обробки викликів CS виконує різні дії, необхідні для встановлення з'єднання між двома абонентами. Як тільки ідентифікатори як абонента, що викликає, так і викликаного, відомі, CS дає вказівку відповідним контролерам шлюзу встановити шлях між абонентами з використанням комутаційної мережі.

Використання підходу з урахуванням кластера Call Server має низку переваг проти традиційної незалежної телекомунікаційної мережі з урахуванням комутаторів. До таких переваг можна віднести такі:

- подання єдиного логічного комутатора та зниження витрат на експлуатацію та управління - OAM (operations and management). Система на основі кластера серверів викликів включає кілька серверів викликів, які функціонують як єдина логічна система. Такий кластер можна використовувати для обслуговування більшої географічної області, ніж та, яку нині обслуговує одна звичайна CS. Це значно знижує вартість OAM, що становить великий інтерес для постачальника послуг телекомунікації;

- масштабованість. Кластер CS може бути організовано за допомогою невеликої кількості серверів обробки викликів, а додаткові сервери обробки викликів можуть додаватися поступово зі збільшенням потреби в ємності;

- відмовостійкість (включаючи географічну надмірність). Оскільки будь-який виклик може бути оброблено будь-яким CS в кластері, то лінії, оброблювані конкретним CS, можуть бути розподілені між іншими серверами обробки викликів у кластері в разі збою одного з серверів обробки викликів. Якщо сервери обробки викликів розміщено досить далеко один від одного, кластер CS може коректно впоратися з відмовою сервера обробки викликів через стихійне лихо або терористичну атаку;

- розподіл навантаження. Сучасні телекомунікаційні мережі, в яких розгорнуто кілька незалежних серверів обробки викликів, починають відхиляти спроби викликів, коли поточний обсяг викликів на даній CS стає занадто великим для задоволення вимог до рівня обслуговування, або коли використання ключового системного ресурсу перевищує заданий поріг. У кластері CS замість

відхилення виклик можна направити на інший слабозавантажений сервер викликів, у якого є вільна потужність для перемикання додаткових викликів.

Кластеризація серверів викликів – цікава концепція. Важливими є низка питань для кластеризації серверів викликів. До них належать: вибір сервера обробки викликів, вибір топології для підключення серверів викликів у кластері, розподіл навантаження між серверами викликів, білінг та технічне обслуговування.

Існує ряд стратегій вибору Call Server, що базуються на характеристиках виклику [38]. Сервер викликів, якому належать дані про стан ініціатора виклика, вибирається для обробки нового виклика. Дані про стан абонента – це дані, що змінюються з незайнятого на зайнятий, коли абонент бере участь у виклику. Після надходження виклику необхідно отримати доступ до даних стану сервера, що викликає. Оскільки віддалений доступ до даних обходиться дорожче, ніж локальний, для комутації виклику вибирається клас методів, у яких працює Call Server, що зберігає дані про стан абонента. Після того, як дані про набір номера зібрано, стають відомі дані абонента для завершення обробки виклика.

Слід зазначити три методи доступу до даних абонента: віддалений доступ, реплікація і міграція. Ці методи працюють однаково, якщо дані про стан абонента є локальними для сервера викликів.

Відмінності виникають тоді, коли дані про стан абонента, що викликається, доступні на віддаленому сервері викликів. При дистанційному доступі до даних про стан кожного абонента процес здійснюється дистанційно. У разі реплікації дані про стан абонента копіюються при перемиканні (обробці) виклику Call Server. Кожна наступна операція читання виконується локально, тоді як повідомлення на віддалений сервер обробки викликів надсилаються для забезпечення узгодженості даних під час операції запису. Дані про стан абонента, що викликається, переміщаються на сервер викликів, що перемикається (обробляється) виклик при використанні стратегії міграції. Після завершення виклика дані про стан переміщаються назад на сервер викликів, де спочатку зберігалися.

Відносна продуктивність політик залежить від кількості звернень до даних за виклик й відношень читання/запису.

У роботі використано віддалену політику для доступу до даних про виклики та стан абонентів, що викликаються. При цьому політика управління даними не сильно впливає на відносну продуктивність методів розподілу навантаження.

Одним із прикладів кластеризації серверів можна назвати розробку компанії Cisco. Це кластеризація CallManager для IP-телефонів [39]. Cisco CallManager — це програмний компонент обробки викликів для вирішення корпоративної IP-телефонії Cisco. Місткість одного сервера на базі CallManager складає всього 7500 IP-телефонів. Кластеризація Cisco CallManager може забезпечити масштабування до 30 000 IP-телефонів на кластер. Абоненти статично призначаються диспетчерам викликів, які виконують комутацію. Хоча система і забезпечує стійкість до відмови, проте динамічний розподіл навантаження не обробляється системою. В інших комерційних некластерних системах виклик відхиляється при виникненні перевантаження.

3.5 Структурна модель Call Server cluster

Модель Call Server cluster представлено основними компонентами: загальною пам'яттю (Shared Memory), елементами обробки PE (Processing Element) та елементами обробки введення-виведення IOP (Input/Output Processor).

Окрім основного призначення, загальна пам'ять виконує функції модуля управління, який повністю адресується кожним процесором, і дані у пам'яті спільно використовуються всіма процесорами. Робота загальної пам'яті носить динамічний характері і дозволяє декількома елементами обробки одночасно читати чи записувати одні й ті ж самі комірки пам'яті. Мережа, яка дозволяє будь-якому IOP або PE отримати доступ до будь-якого модуля пам'яті, називається Interconnect.

Процесорний елемент PE включає ЦП, локальну пам'ять та кеш. Ряд процесорних елементів забезпечує обчислювальну потужність загального призначення, окрім функцій введення/виведення. У кластері використовується політика планування з розподілом у часі. Всі PE функціонально еквівалентні і виконують те саме програмне навантаження. Програма досягає результату незалежно від того, на якому PE вона працює. Спеціального призначення якогось PE під конкретне завдання не існує. Будь-який PE може виконати будь-яке завдання. Перевага такого підходу в тому, що всі PE завантажуються однаково, а обчислювальне навантаження розподіляється природним чином.

Для створення кластера Call Server кілька Call Servers з'єднуються через швидкісну мережу. Весь зв'язок між різними серверами викликів обробляється через IOP кожної CS. Кластер із трьох серверів викликів показано на рис. 3.1.

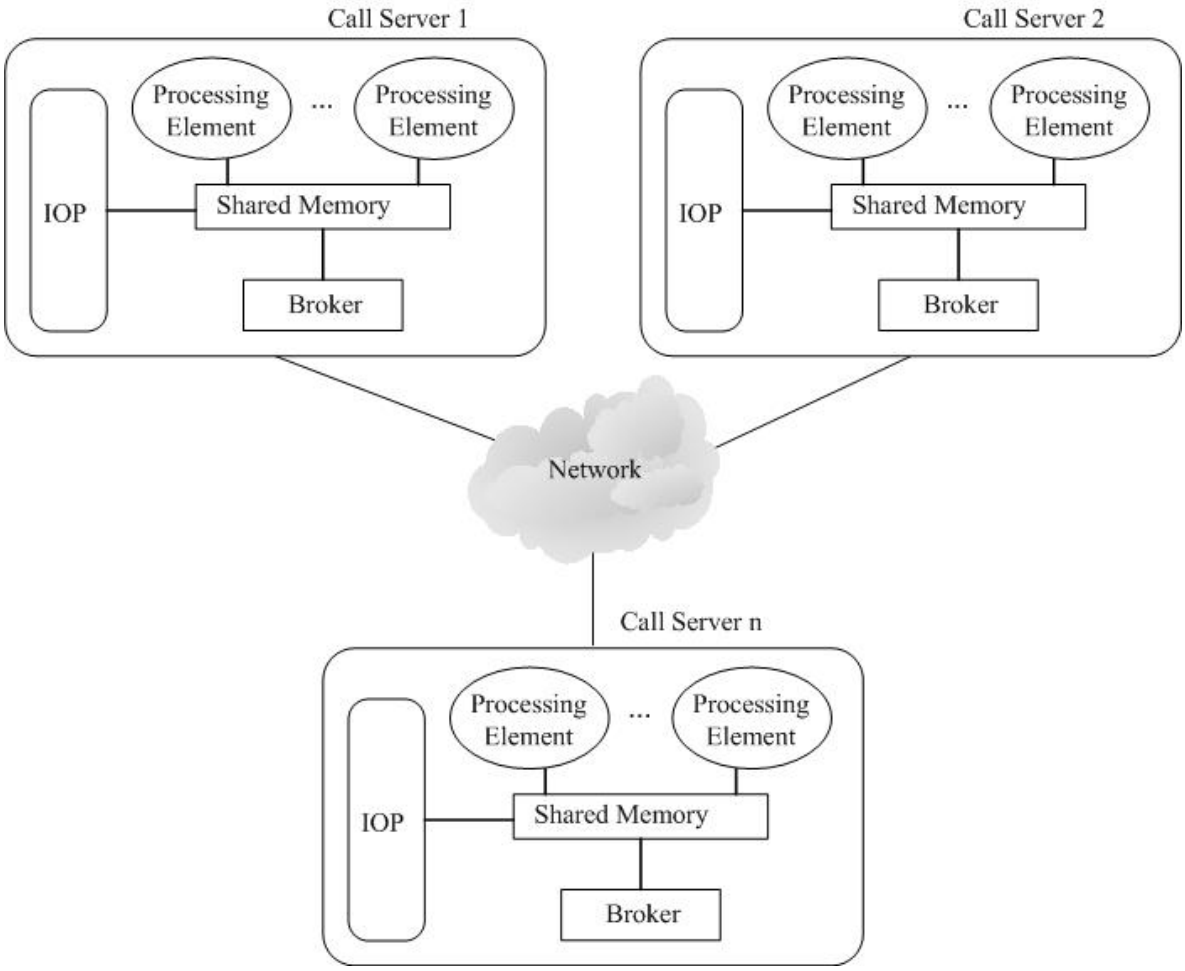

Рисунок 3.1 - Модель Call Server cluster

Інтерфейси введення/виведення забезпечуються 1/0 елементами обробки IOP. Ці пристрої забезпечують два типи зв'язку: канали зв'язку Optical Carrier та інші зовнішні високошвидкісні комунікаційні інтерфейси, такі як Ethernet та X.25.

Брокер - це незалежний процес, подібний до IOP, він обробляє нові вхідні виклики в міру їх надходження. Брокер необхідний для забезпечення функціональності розподілу навантаження. Процес роботи брокера в архітектурі Call Server використовується для керування розподілом навантаження та обміну інформацією про навантаження. Коли навантаження на сервер викликів перевищує наперед заданий поріг, брокер намагається знайти відповідний приймаючий CS, який працює з розрахунковою потужністю. Після визначення приймаючого CS наступним найважливішим кроком є визначення того, яке навантаження має бути передана приймаючим CS для запуску брокера.

3.6 Індекс навантаження як характеристика продуктивності сервера

Існують різні можливі індикатори навантаження на сервери викликів, які включають час відгуку для вхідних викликів або середнє завантаження процесора сервером викликів. Для аналізу навантаження у якості індикатора навантаження може використовуватися розмір черги, що очікує, що пов'язано з балансуванням навантаження.

Для аналізу продуктивності сервера використано характеристику індексу навантаження. Цей показник пов'язаний із продуктивністю всієї системи та легко обчислюється. Крім того, він визначається під час відгуку системи. Час відповіді на виклик вимірюється як затримка вихідного виклику, тобто затримка ініціації. Затримка ініціації - це час очікування нового повідомлення про виклик для досягнення підсистеми обробки сервера викликів.

З точки зору вхідного трафіку для кластера Call Server найкращим індикатором є затримка 95-го процентілю - 95-й POD (percentile origination delay) - P_{POD}, яка використовується для виявлення перевантаження.

У роботі для серверів викликів у якості індекса навантаження вибирається P_{POD}. Використання цього параметра засновано на тому факті, що більшість технічних характеристик телефонних комутаторів та серверів виражаються у вигляді P_{POD} [40].

Крім того, цей показник часто використовується Інтернет-провайдерами та дата-центрами як стандартний метод вимірювання пропускної спроможності каналу для оцінки трафіку при операціях білінгу. Наприклад, цей метод дозволяє користувачам різко збільшувати трафік протягом 5 % часу без збільшення вартості оплати послуг дата-центру. Більшість Інтернет-провайдерів використовують п'ятихвилинний інтервал вибірки для запису трафіку, а потім розраховують використання по 95-му процентілю.

Таким чином, показник P_{POD} використано у роботі як величина стандартного або прогнозованого відхилення навантаження на сервер, тобто решта 5 % трафіку є піковими. Тому короткі періоди з дуже високим трафіком, можливо, в десятки разів вище, ніж стандартне навантаження, в роботі не досліджуються.

3.7 Аналіз режимів переадресації одиночного CS у кластері

Вхідні завдання, підписані на систему, спочатку направляються диспетчеру навантаження або брокеру, який відстежує найслабше завантажений серверний модуль у системі. При підписці на нове завдання вона буде відправлена на серверний модуль із найменшим навантаженням. Таким чином, спочатку досягається статичне балансування навантаження.

Далі починає роботу балансувальник навантаження, який відповідає за прийняття рішень щодо динамічного перенесення завдань між серверами. Кожен підхід до балансування навантаження враховує два основні аспекти: метод оцінки навантаження та політику міграції навантаження. Процес розподілу навантаження, що виконується брокером, засновано на блок-схемі, яку показано на рис. 3.2.

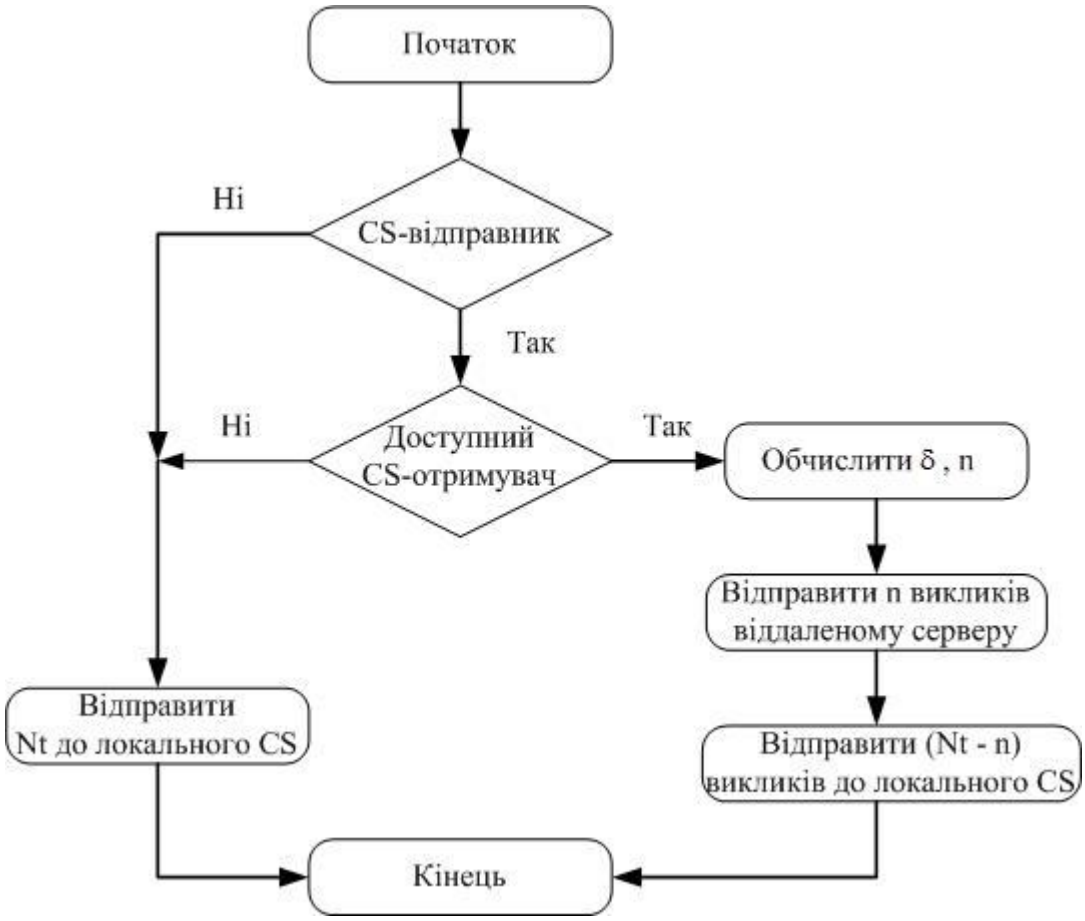

Рисунок 3.2 - Алгоритм розподілу навантаження

На схемі відображено такі показники: N_t - загальна кількість доступних викликів на сервері обробки викликів, які можуть бути переадресовано на

дистанційний одержувач CS; δ - коефіцієнт переадресації, який являє собою частку від загальної кількості нових викликів, які повинні бути відправлені на віддалений CS. Таким чином, кількість нових викликів, відправлених на віддалений сервер, визначається виразом:

$$n = \delta \cdot N_t . \qquad (3.1)$$

Особливості роботи блок-схеми:

- крок 1: щоразу, коли надходить новий вхідний виклик, брокер вирішує, чи є поточний CS вузлом-відправником чи ні (політика передачі);

- крок 2: якщо поточний CS є вузлом-відправником, то він шукає відповідний CS-отримувач, на який він може передати навантаження про нові вхідні виклики (політика міграції);

- крок 3: після успішного виявлення CS-одержувача брокер застосовує політику переадресації для визначення значень δ та n;

- крок 4: n викликів передаються на віддалений CS-одержувач, а решта викликів ($N_t - n$) обробляються на локальному CS;

- крок 5: якщо локальний CS не є вузлом-відправником або немає доступного CS-отримувача, всі вхідні виклики обробляються на локальному CS.

У роботі проведено дослідження режимів переадресації, суть яких засновано на ідеї перенаправлення заданої частки нових вхідних викликів з Call Server на слабонавантажений CS. Проаналізовано адаптивні режими переадресації. Дослідження проведено з метою оцінки ступеня стану системи, який направлено для досягнення високої продуктивності.

Існує кілька адаптивних режимів переадресації, які відрізняються способом розрахунку кількості переадресованих дзвінків з використанням інформації про завантаження відправляючого та/або приймаючого сервера викликів. У цих режимах використовуються різні ступені стану системи:

- адаптивний режим 1 використовує інформацію про завантаження відправляючого CS (Ending CS);

- адаптивний режим 2 використовує інформацію про завантаження як відправляючого CS, так і отримуючого CS, але максимальна кількість вхідних викликів, які можуть бути перенаправлені на віддалений CS, фіксована (Fixed sending/receiving CS);

- адаптивний режим 3 використовує інформацію про завантаження приймаючого CS (Receiving CS);

- адаптивний режим 4 використовує інформацію про завантаження як відправляючого CS, так і отримуючого CS, але максимальна кількість вхідних викликів, які можуть бути перенаправлені на віддалений CS, є змінним параметром (Variable sending/receiving CS).

Статичний режим переадресації у роботі не розглядався з огляду на те, що цей режим не використовує жодної інформації про завантаження ні відправляючого CS, ні одержуючого CS при визначенні δ. Цей параметр у свою чергу утримується на фіксованому значенні. При цьому спостерігається передача фіксованої частки вхідних викликів від загальної кількості викликів на приймаючий сервер та чекають обробки модулем обробки сервера викликів. Тому при високих навантаженнях дослідження даного режиму не становить інтересу.

3.7.1 Аналіз режиму переадресації на основі завантаження відправляючого одиночного CS у кластері

У цьому адаптивному режимі 1 (Ending CS) використовано інформацію про завантаження відправляючого CS та враховується стан завантаження відправляючого сервера викликів для розрахунку коефіцієнта переадресації δ. Складною частиною аналізу є визначення коефіцієнта переадресації, виходячи з додаткового навантаження відправляючого CS. Це пов`язано з нелінійною поведінкою сервера викликів, що він працює поза межами проєктної потужності E. Дані залежності індексу навантаження від інтенсивності навантаження на одиночний CS кластеру відображено в таблиці 3.1.

Таблиця 3.1 – Результати залежності індексу навантаження від інтенсивності навантаження на одиночний CS у кластері

Інтенсивність навантаження, викл/год	500	1000	1500	2000	2500	3000	3200	3250
Індекс навантаження, мс	20	45	60	76	90	95	1000	1600

Розрахункова потужність сервера обробки викликів визначається як інтенсивність навантаження, з якою може впоратися сервер обробки викликів з менше певного заданого значення (140 мс для моделі CS, що розглядається).

На рис. 3.3 показано залежність індексу навантаження від інтенсивності навантаження на одиночний CS в кластері.

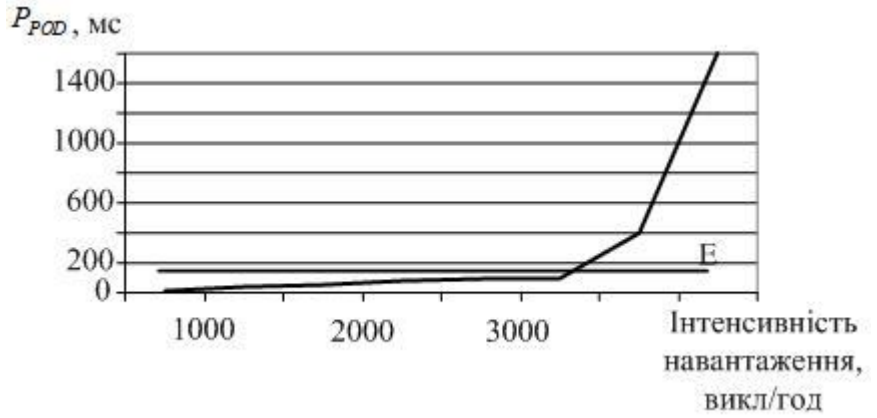

Рисунок 3.3 - Залежність індексу навантаження від інтенсивності навантаження на одиночний CS у кластері

З рис. 3.4 видно, що розрахункова потужність сервера обробки викликів може обробити 3157 викл/год. Продуктивність системи різко погіршується, коли вхідне навантаження виходить за межі проєктної потужності. Для детального аналізу зміни P_{POD} поза межами проєктної потужності показана докладна частина кривої (рис. 3.4).

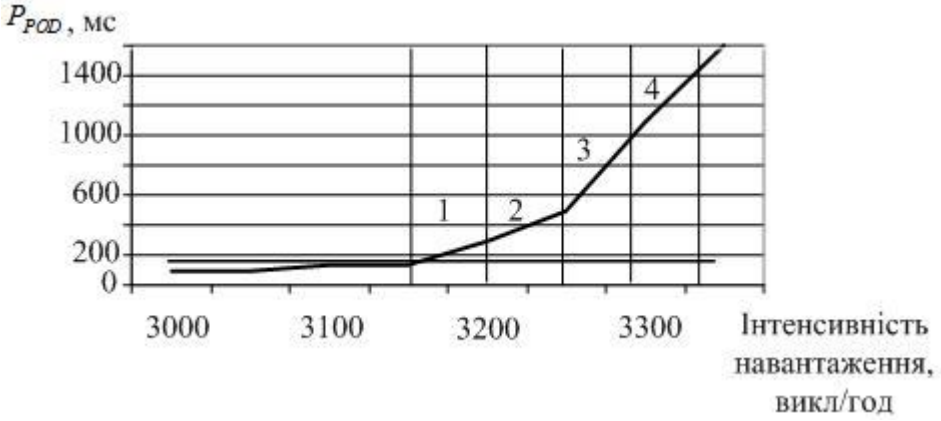

Рисунок 3.4 - Залежність індексу навантаження від інтенсивності навантаження на одиночний CS у кластері за межами проєктної потужності

Значення P_{POD}, що дорівнює 1600 мс, вважається дуже великим. При цьому кластер на базі одного Call Server досягає межі при інтенсивності вхідного навантаження 3327 викликів/год.

Якщо розглянути лінійну залежність між індексом навантаження і часткою переадресованих викликів, то CS-відправник може переадресовувати велику кількість викликів на віддалений CS. Це може спричинити додаткові накладні витрати та може призвести до перевантаження віддаленого сервера обробки викликів.

Щоб уникнути цієї ситуації, нелінійну частину кривої поза межами проєктної потужності розділено на невеликі лінійні ділянки. Це чотири ділянки, які позначено як 1, 2, 3, 4 на рис. 3.5. Діапазони інтенсивності навантаження та відповідні їм значення P_{POD} для граничних крапок показано у таблиці 3.2.

Таблиця 3.2 - Діапазони інтенсивності навантаження та відповідні їм значення P_{POD}.

Ділянка	Інтенсивність навантаження, викл/год	Зміна індексу навантаження, мс
1	3157-3199	140 - 230
2	3199-3241	230 - 440
3	3241-3283	440 - 850
4	3283-3325	850 - 1600

Аналіз лінійних ділянок показує зв'язок між δ та індексом навантаження перевантаженого CS, що відображено на рис. 3.5.

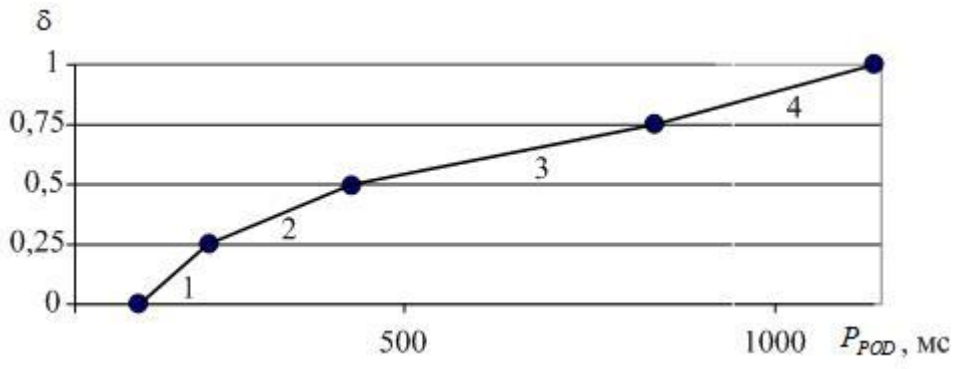

Рисунок 3.5 - Залежність між δ та індексом навантаження P_{POD} відправляючого CS

Значення δ обчислюється з використанням рівняння прямої цих лінійних ділянок - 1, 2, 3, 4. Видно, що з визначення максимального значення δ для конкретної ділянки можливий діапазон значень δ (0 ÷ 1) розбивається на чотири рівні частини. Максимальні значення δ відповідне ділянкам 1, 2, 3 і 4, встановлюються рівним 0,25, 0,5, 0,75 й 1,0 відповідно.

Якщо індекс завантаження L_S відправляючого Call Server знаходиться на ділянці 1, то значення δ обчислюється з використанням рівняння:

$$\delta_1 = \frac{0{,}25(L_S - 140)}{(230 - 140)}. \tag{3.2}$$

Значення 140 та 230 відповідають P_{POD} у двох кутах ділянки 1. Аналогічно рівняння для ділянок 2, 3 та 4 відображено у формулах:

$$\delta_2 = 0{,}25\left[1 + \frac{(L_S - 230)}{(440 - 230)}\right], \tag{3.3}$$

$$\delta_3 = 0{,}25\left[2 + \frac{(L_S - 440)}{(850 - 440)}\right], \tag{3.4}$$

$$\delta_4 = 0{,}25\left[3 + \frac{(L_S - 850)}{(1600 - 850)}\right]. \tag{3.5}$$

Залежно від того, на якій ділянці знаходиться індекс завантаження CS, для обчислення значення δ використовується відповідне рівняння.

Розрахунки показали, що коефіцієнт переадресації δ приймає такі значення:

- δ_1 змінюється в межах 0,0028÷0,25;
- δ_2 змінюється в межах 0,2512÷0,5;
- δ_3 змінюється в межах 0,5006÷0,75;
- δ_4 змінюється в межах 0,7503÷1,0.

Аналіз рис. 3.6 показав, що δ збільшується зі збільшенням навантаження відправника, а також враховує нелінійний вплив інтенсивності навантаження показника навантаження. Швидкість збільшення коефіцієнта переадресації

зменшується в міру того, як навантаження на відправляючий сервер викликів стає вищим.

Визначивши δ, розраховується точна кількість викликів n для переадресації:

$$n = \delta \cdot N_t \cdot P, \tag{3.6}$$

де P - константа, що відповідає максимальній частці переадресованих вхідних викликів, які можуть бути переведено на віддалений сервер викликів. У цьому режимі P підтримується лише на рівні 40 %. Результати розрахунку при різних значеннях δ показано на рис. 3.6.

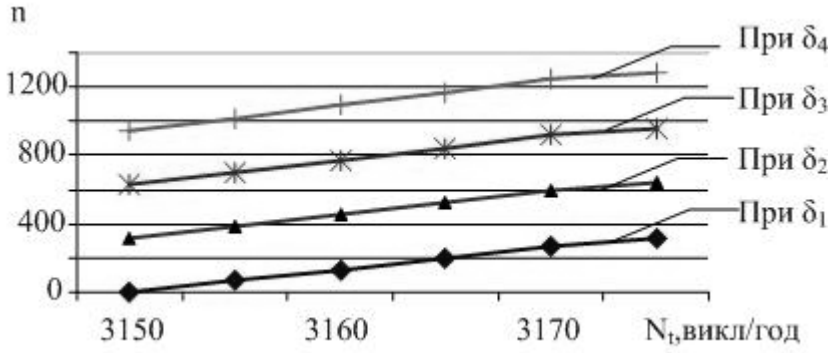

Рисунок 3.6 – Результати розрахунку n при різних значеннях δ

Очевидно, що зі збільшенням кількості викликів на сервері кількість викликів для переадресації буде збільшуватися.

3.7.2 Аналіз режиму переадресації на основі завантаження відправляючого та приймаючого одиночного CS у кластері з фіксованою кількістю викликів

Цей адаптивний режим 2 (Fixed sending/receiving CS) використовує інформацію про завантаження як відправляючого, так і приймаючого серверов викликів для розрахунку n. При цьому використовується динамічне значення P замість фіксованого. При розрахунку δ використовується той самий алгоритм з урахуванням інформації про завантаження відправляючого CS. Залежно від стану завантаження CS-приймача пропонуються чотири рівні готовності, які відповідають резервній ємності CS-приймача.

Якщо індекс завантаження приймаючого CS позначити L_r, то доступна потужність A приймаючого CS визначається як різниця між порогом політики міграції TH_L та L_r:

$$A = TH_L - L_r. \tag{3.7}$$

Поріг політики міграції TH_L є фіксованим значенням та близьким до показника навантаження, що відповідає проєктній потужності Call Server. Сервер викликів призначається вузлом-отримувачем, якщо індекс навантаження цього сервера викликів нижче за поріг політики міграції. При цьому визначено чотири рівня готовності резервної ємності CS одержувача. Використовується набір із чотирьох значень P, причому $P_1 < P_2 < P_3 < P_4$, які відповідають одному з рівнів готовності CS-приймача:

- для P_1:

$$0 \le A \le (0{,}125 \cdot TH_L); \tag{3.8}$$

- для P_2:

$$(0{,}125 \cdot TH_L) \le A \le (0{,}25 \cdot TH_L); \tag{3.9}$$

- для P_3:

$$(0{,}25 \cdot TH_L) \le A \le (0{,}5 \cdot TH_L); \tag{3.10}$$

- для P_4:

$$(0{,}5 \cdot TH_L) \le A \le TH_L. \tag{3.11}$$

Значення рівнів готовності резервної ємності CS одержувача наведено в таблиці 3.3.

Таблиця 3.3 - Значення рівнів готовності резервної ємності CS одержувача

Рівні готовності резервної ємності CS одержувача	Досяжна потужність, мс
P_1	$0 < A \leq 17,5$
P_2	$17,5 < A \leq 35$
P_3	$35 < A \leq 70$
P_4	$70 < A \leq 140$

Як тільки рівень доступності приймаючого CS визначено, вибирається відповідне значення P для визначення кількості переадресованих викликів за формулою (3.6). Результати розрахунку n при різних значеннях δ та P показано на рис. 3.7.

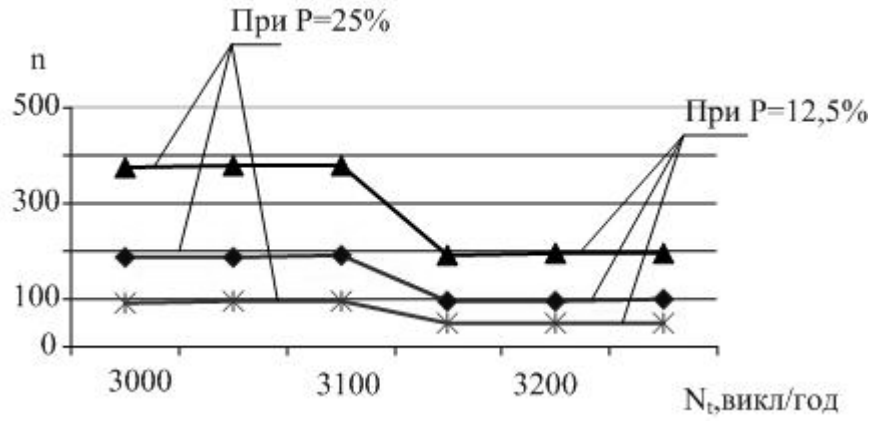

Рисунок 3.7 - Результати розрахунку n при різних значеннях δ та P

Таким чином, для визначення n використовується інформація сервера обробки викликів відправника та одержувача: навантаження сервера обробки викликів відправника використовується для визначення δ, а навантаження сервера обробки викликів одержувача використовується для визначення P.

Відмінність даного режиму від інших адаптивних режимів полягає у тому, що P є динамічною величиною, яка вибирається з безлічі рівнів готовності CS-приймача. Перевага такого підходу полягає в тому, що якщо CS одержувач працює на межі проєктної потужності, то відправляючий CS знизить свою максимальну межу P, щоб адаптувати значення δ на основі навантаження як одержувача, так і відправника. Але якщо одержувач працює з низьким навантаженням, то відправляючий CS буде використовувати більш високі

значення максимального ліміту P, що призведе до більш ефективної передачі нових викликів на віддалений Call Server.

3.7.3 Аналіз режиму переадресації на основі завантаження приймаючого одиночного CS у кластері

Цей адаптивний режим 3 (Receiving CS) використовує інформацію про навантаження приймаючого сервера викликів тільки для розрахунку коефіцієнта переадресації δ. Алгоритм роботи засновано на більш точному співвідношенні між P_{POD} та інтенсивністю навантаження в порівнянні з попередніми режимами. На рис. 3.8 показано залежність індексу навантаження від інтенсивності навантаження на одиночний CS кластері.

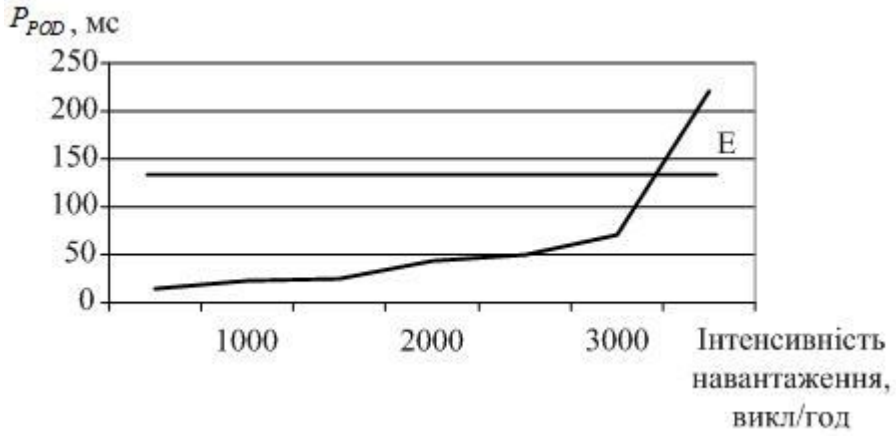

Рисунок 3.8 - Залежність індексу навантаження від інтенсивності навантаження на одиночний CS у кластері

До P_{POD} тривалості 70 мс залежність між індексом навантаження та інтенсивністю навантаження є приблизно лінійною. Тому δ визначається за лінійною залежністю, якщо індекс завантаження CS приймача менше 70 мс. Таким чином, коефіцієнт переадресації можна обчислити за допомогою лінійної інтерполяції, яка визначається виразом:

$$\delta = \frac{A}{TH_L}. \qquad (3.12)$$

Якщо індекс завантаження CS приймача знаходиться в діапазоні від 70 до 140 мс, то приймач CS перевантажується. Щоб уникнути цього, невелика фіксована частка викликів підлягає переадресації, коли індекс завантаження перевищує 70 мс.

Надалі доцільне визначення згідно з алгоритмом, що показано на рис. 3.9.

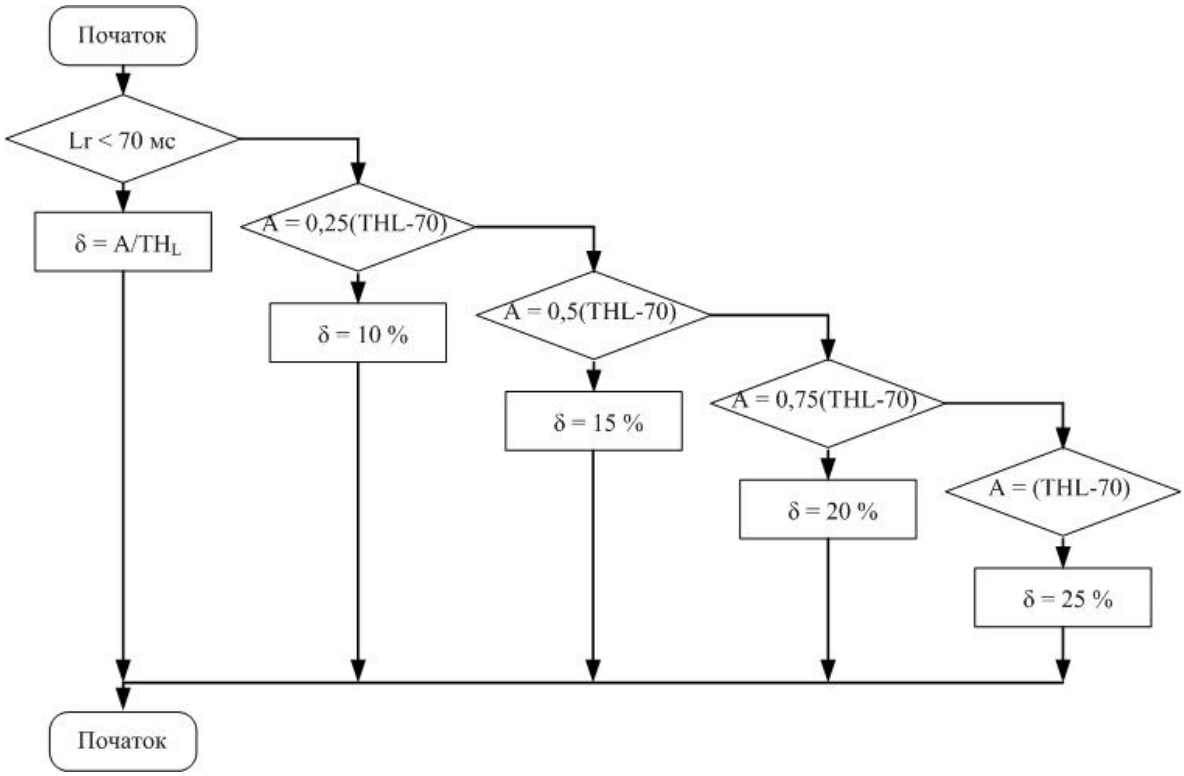

Рисунок 3.9 - Алгоритм визначення δ для режиму переадресації на основі завантаження приймаючого CS

Результати розрахунку n при різних значеннях δ показані на рис. 3.10.

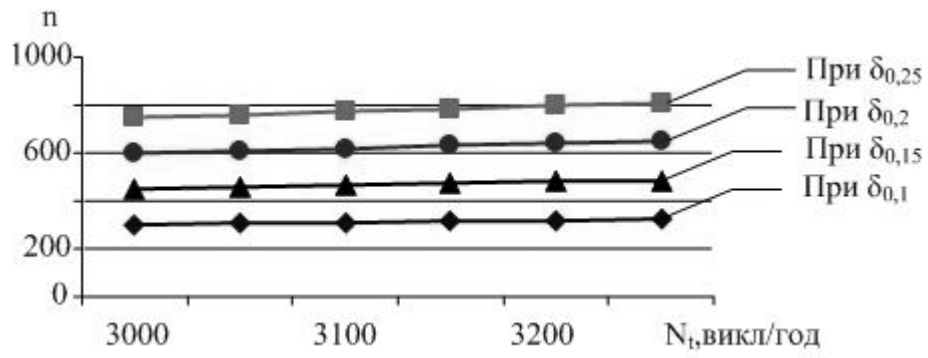

Рисунок 3.10 - Результати розрахунку n при різних значеннях δ.

3.7.4 Аналіз режиму переадресації на основі завантаження відправляючого та приймаючого одиночного CS у кластері зі змінною кількістю викликів

При аналізі адаптивного режиму 4 (Variable sending/receiving CS) використовується інформація про завантаження як відправляючого, так і отримуючого CS, але максимальна кількість вхідних викликів, які можуть бути перенаправлені на віддалений CS, є змінним параметром. Коефіцієнт переадресації обчислюється з урахуванням інформації про навантаження як відправляючого CS, так й приймаючого CS, незалежно один від одного. Оскільки добуток δ і P може мати максимальне значення, що дорівнює 1, то передбачається, що компоненти відправника й одержувача роблять однаковий внесок у визначення n. Таким чином, кількість викликів, що підлягають переадресації, задається виразом:

$$n = \Delta \cdot N_t, \tag{3.13}$$

де $\Delta = 0{,}5 \cdot \delta_1 + 0{,}5 \cdot \delta_2$. За такого розрахунку δ_1 визначається з використанням режиму на основі стану завантаження CS-відправника, а δ_2 розраховується з використанням стану завантаження CS-приймача.

Загальний розкид значень δ_1 становить 0,0028÷1, а визначення δ_2 виконується згідно з алгоритмом на рис. 3.10. Проміжні розрахунки становлять: 0,175; 0,175; 0,325; 0,625. З урахованням отриманих даних характеристики робочого процесу показано на рис. 3.11.

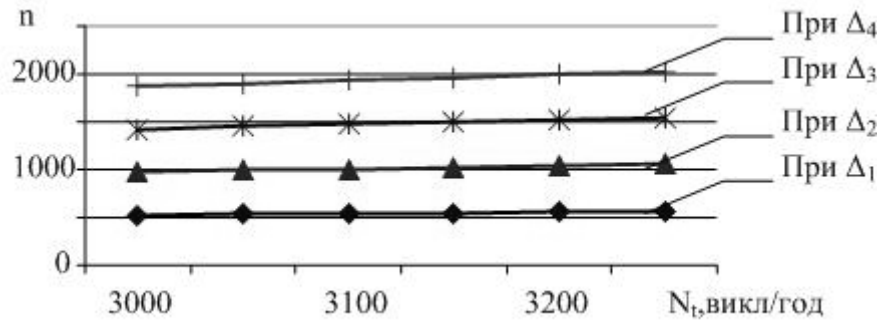

Рисунок 3.11 – Результати розрахунку n при різних значеннях Δ

Таким чином, проведено дослідження відносних характеристик адаптивного підходу для ефективної обробки стрибків навантаження, що виникають у кластері. Адаптивні режими, які застосовуються залежно від поточного стану системи, працюють краще в порівнянні зі статичним режимом, для якого тільки фіксована частина нових вхідних викликів переадресовується з перевантаженого на недовантажений CS. Режим Fixed sending/receiving CS, як і режим Receiving CS, які використовують інформацію про навантаження від одержувача CS, демонструють хорошу продуктивність для широкого діапазону параметрів робочого навантаження. Проте режим Receiving CS вважається кращим вибором, оскільки він використовує менше інформації про стан системи, ніж режим Fixed sending/receiving CS, і тому очікується, що накладні витрати будуть нижчими.

4 ДОСЛІДЖЕННЯ ХАРАКТЕРИСТИК CS У КЛАСТЕРІ

4.1 Аналіз моделі клієнт-сервер для сервера викликів

Для аналізу характеристик CS у кластері розглянуто три ієрархічні рівні або домени: мережна модель, модель вузла та модель процесу.

Мережна модель визначає загальний обсяг моделі, що моделюється: об'єкти в системі, а також їх фізичне розташування, взаємозв'язки та конфігурації. Модель може містити одну підмережу, один вузол або безліч взаємопов'язаних вузлів й підмереж. Основними структурними блоками домену мережі є підмережі, вузли та канали зв'язку. У цьому розумінні один сервер виклику становить об'єкт підмережі в домені мережі. Кластер створюється в мережному домені шляхом з'єднання кількох серверів викликів через канали зв'язку.

Модель вузла надає детальну інформацію про окремий вузол або підмережу. Основними елементами моделі вузла є процесори, модулі передавача та приймача. Шляхи зв'язку між базовими елементами моделі вузла забезпечуються потоками пакетів. Різні компоненти, такі, як периферійний модуль, процес введення/виведення або система обробки CS, розглядаються в цьому контексті як модель процесора.

Модель процесу визначає поведінку процесів і черг, існуючих в моделі вузла. Детальна поведінка процесу контролюється за допомогою процедур введення/виведення або перехідних рівнів.

Розроблена модель клієнт-серверу складається із двох основних підмоделей. Ці підмоделі призначені для периферійного модуля (PM) та основного сервера викликів. Ці підмоделі становлять клієнт-серверну модель. Клієнти представляють процеси, що використовуються для моделювання периферійних модулів, а процеси представляють сервери моделі клієнт-сервер. Загальний огляд моделі клієнт-сервер представлено на рис. 4.1.

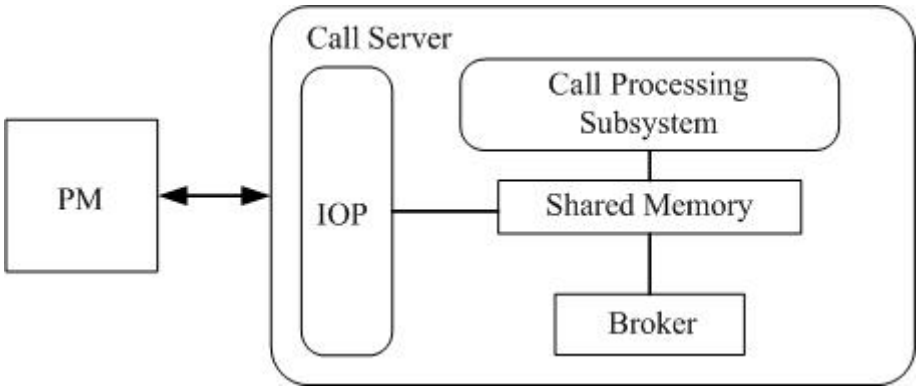

Рисунок 4.1 - Модель клієнт-сервер для сервера викликів

На рис. 4.1 показані такі компоненти моделі: загальна пам'ять (Shared Memory), підсистема обробки викликів (Call Processing Subsystem), елементи обробки введення-виведення (IOP), периферійний модуль (PM). Периферійний модуль працює для звичайного телефонного зв'язку, що використовується лише для голосових викликів. Час між вхідними викликами ґрунтується на експоненційному розподілі із заданим середнім часом між викликами.

Отримавши нове повідомлення про вхідний виклик, периферійний модуль створює незалежний процес, який виступає в ролі незалежного клієнта, що взаємодіє з Call Server. Кожен периферійний модуль належить одному серверу викликів. Периферійний модуль може надсилати повідомлення лише своєму серверу обробки викликів за замовчуванням, але він також може отримувати повідомлення від віддалених серверів обробки викликів.

Серед особливостей PM слід відзначити можливість безпосередньо зв'язуватися з будь-яким із віддалених серверів обробки викликів у кластері. Тому кожен клієнт у периферійному модулі повинен зберігати ідентифікаційний ключ інформації про стан виклику, що зберігається на віддаленому сервері викликів. Цей ідентифікатор передається з кожним повідомленням (відповіддю або повідомленням про вихід), що надсилається з цього периферійного клієнтського модуля на віддалений Call Server. Видалений сервер викликів використовує цей ідентифікатор для виявлення інформації про стан виклику, для якого було надіслано повідомлення.

Обробка виклику периферійним модулем пояснюється діаграмою станів. На рис. 4.2 показано діаграму станів обробки Call Server робочого процесу периферійного модуля. Ця діаграма станів дійсна як для відправника, так й для одержувача виклику.

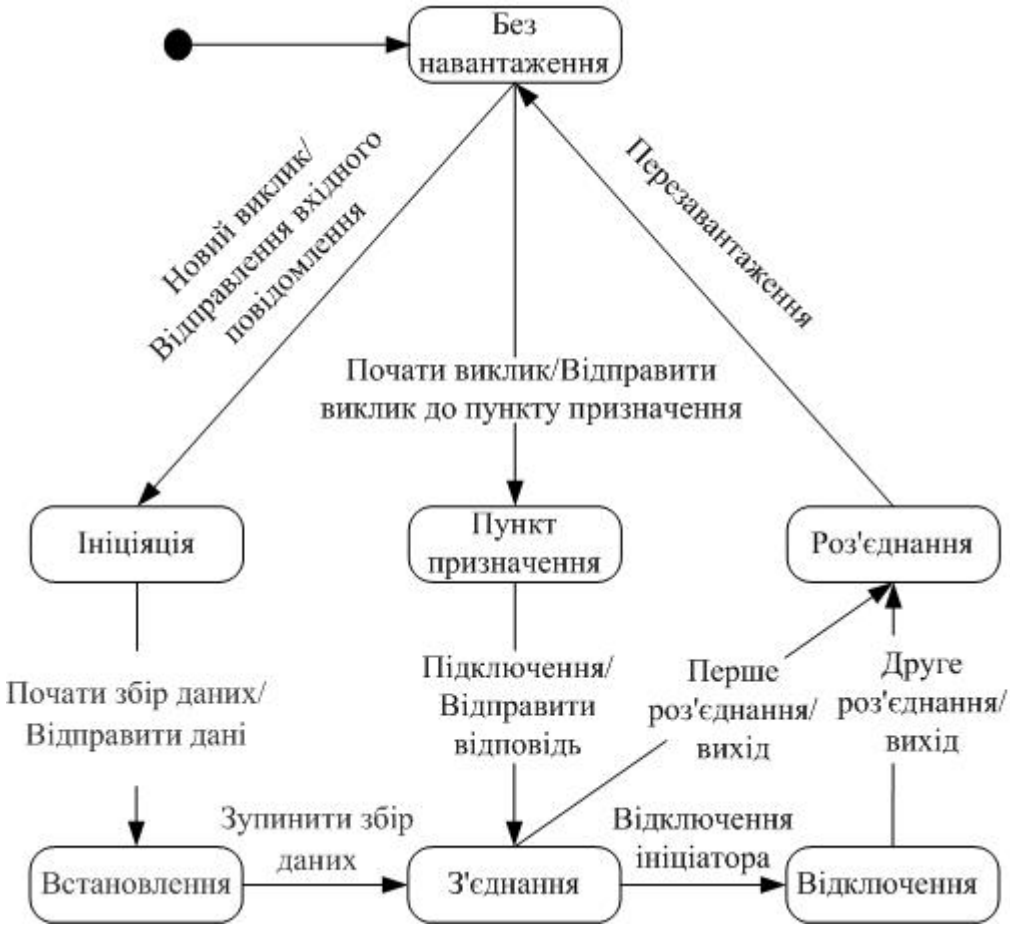

Рисунок 4.2 - Діаграма станів для обробки виклику периферійного модуля

Як ініціатор, периферійний модуль надсилає повідомлення "Джерело" своєму CS, а потім чекає відповіді в стані "Джерело". При отриманні повідомлення "Початти збір даних" він починає збирати дані та надсилає їх на CS. Як тільки дані відправлено, він очікує в стані "Встановлення" для встановлення виклику за допомогою CS. Як тільки процес встановлення виклику завершено, периферійний модуль отримує повідомлення "Зупинити збирання даних", і лінія підключається до лінії призначення. Процес PM викликаючого абонента переходить у стан "Підключено" й залишається в стані "Підключено" до тих пір, поки інший учасник виклику також підключений. Як тільки один із учасників виклику вимикається, його периферійний модуль надсилає повідомлення "Вихід" на CS та звільняє ресурси для цього абонента. CS надсилає повідомлення "Вимкнено" на периферійний модуль іншого учасника виклику. Цей периферійний модуль вимикає звук трубки абонента та переходить у стан "Без звуку". Як тільки інший учасник відключається (друге вимкнення), його периферійний модуль також надсилає повідомлення "Вихід" на сервер обробки викликів.

У якості кінцевого абонента периферійний модуль при отриманні повідомлення "Застосувати фізичний дзвінок" посилає сигнал виклику на трубку абонента й переходить у стан "Викликається". Як тільки телефон обслуговується, на CS відправляється повідомлення "Відповідь" і процес PM цього абонента переходить у стан "Підключено". Для різних фаз виклику використовуються різні типи повідомлень. Ключові повідомлення, задіяні в обробці викликів, наведено в Додатку А.

Робочий процес інших частин компонентів для моделі Call Server - моделі процесу введення-виведення, брокера та підсистеми обробки викликів охарактеризовано на діаграмі взаємодії UML, що показано на рис. 4.3.

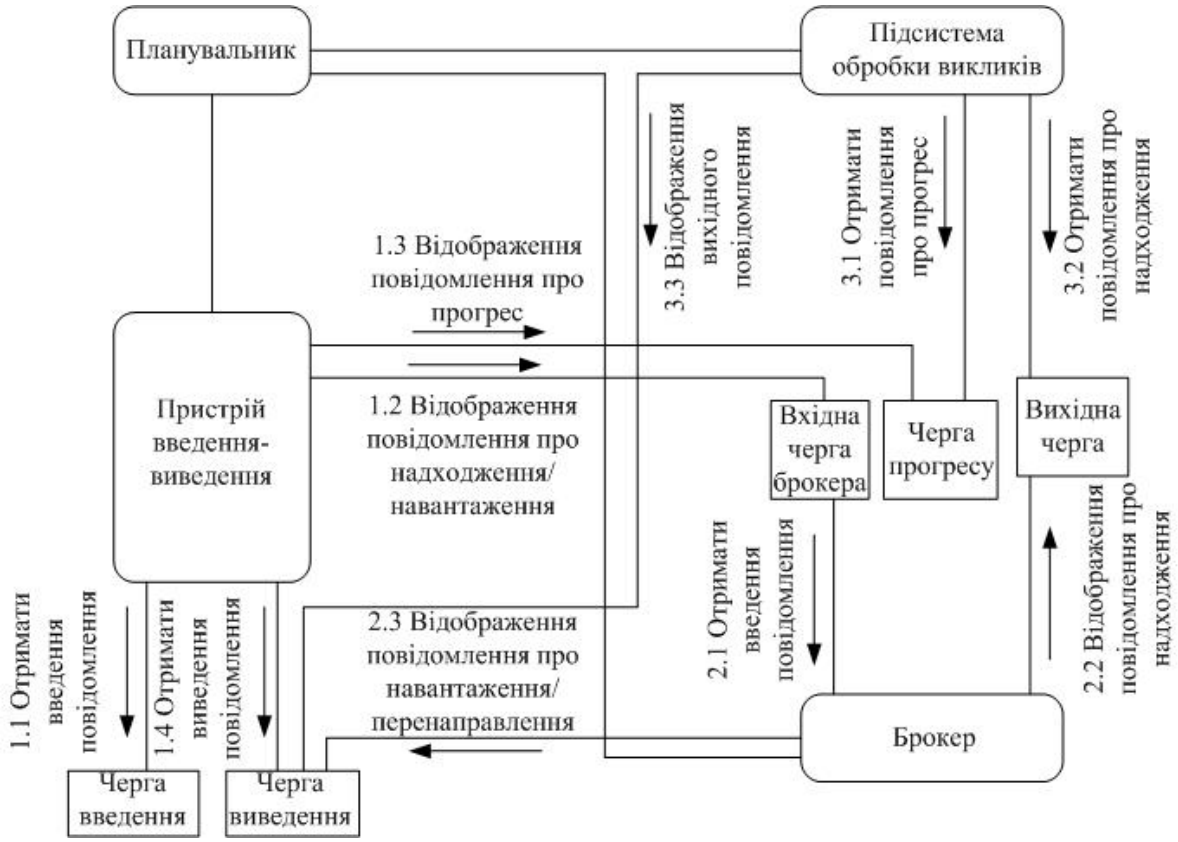

Рисунок 4.3 - Діаграма взаємодії UML для моделі Call Server

Ці компоненти представлені як моделі процесів чи моделі черг. Моделі черг доменного вузла використовуються для моделювання планувальника та різних черг на сервері викликів. Вбудовану модель черги засновано на принципі "першим прийшов - першим обслужений" - FIFO (first in, first out). Стереотипи, що використовуються на цій діаграмі, представлено для характеристики різних процесів та вбудованих черг. Числові мітки в повідомленнях ґрунтуються на

припущенні, що планувальник спочатку викликає процес введення/виведення, а потім процес брокера та процеси підсистеми обробки викликів. Але насправді порядок виклику цих процесів не фіксовано.

Повідомлення, надіслані із зовнішнього модуля, зберігаються у черзі введення. Процес введення/виведення витягує ці повідомлення щоразу, коли він викликається планувальником.

Повідомлення, які мають бути надіслані поза Call Server, містяться в чергу виведення. Підсистема обробки викликів та процес брокера поміщають вихідні повідомлення у чергу виведення. Знову ж таки, процес введення-виведення обробляє ці повідомлення щоразу, коли він викликається планувальником.

Повідомлення про нове надходження та інформацію про завантаження повинні оброблятися процесом брокера в кластері Call Server. Процес введення-виведення розташовує ці повідомлення у вхідну чергу брокера. Брокерський процес витягує ці повідомлення щоразу, коли він викликається планувальником.

Брокерський процес розташовує вихідні повідомлення у вихідну чергу для локальної обробки нового виклику. Повідомлення про відправку очікують у цій черзі обробки підсистемою обробки викликів.

Усі повідомлення, крім вихідних, містяться в чергу прогресу за допомогою процесу введення/виведення. Черга прогресу – це друга вхідна черга для підсистеми обробки викликів.

Модель процесу введення/виведення включає додаткові повідомлення, введені для розподілу навантаження між серверами викликів. Модель може обробляти повідомлення як від віддалених периферійних модулів, так і від віддалених серверів викликів.

Вихідні повідомлення можуть мати кілька адресатів, таких, як локальний периферійний модуль, віддалений периферійний модуль та віддалений сервер обробки викликів. У процесі введення/виведення використовуються тип повідомлення та категорія призначення для ідентифікації точного місця доставки повідомлення.

Повідомлення про завантаження обмінюються між серверами викликів для спільного використання оновленої інформації про стан завантаження кожного сервера викликів. Процес введення/виведення надсилає ці повідомлення на інші сервери викликів. При отриманні інформації про завантаження повідомлення процес введення/виведення ефективно доставляє це повідомлення брокеру для оновлення таблиці бази даних завантаження.

У моделі процесу введення/виведення планувальнику надсилається повідомлення з вимогою РЕ щоразу, коли викликається його переривання. При доступності процесора процес введення/виведення спочатку обробляє нові вхідні повідомлення, а потім вихідні повідомлення. Зрештою, він призначає себе на наступний виклик і надсилає планувальнику повідомлення про звільнення РЕ.

4.2 Вибір параметрів для дослідження

У досліджуваній системі використовуються порогові політики міграції та перенесення. Політика міграції визначає вузол-одержувач, на який передаватимуться виклики з вузла-відправника. Будь-який сервер викликів, індекс навантаження якого нижчий від порогового значення політики міграції, може бути вузлом-одержувачем. Щоб уникнути ситуації, коли кілька вузлів-відправників щоразу вибирають один і той самий вузол-одержувач, для вибору вузла-отримувача з набору вузлів-отримувачів використовується виважена випадкова політика. Ця політика використовує доступну потужність вузла одержувача.

Вузол i з безлічі вибирається з ймовірністю $\frac{A_i}{\sum(A_i)}$, де A_i - доступна потужність вузла i. Політика перенесення ідентифікує вузол-відправник. Коли індекс навантаження сервера викликів перевищує граничне значення політики передачі, він вважається вузлом-відправником. Значення вихідних параметрів представлено у таблиці 4.1.

Таблиця 4.1 – Значення вихідних параметрів для дослідження

Параметр	Значення
1	2
Розмір кластеру	3
Інтенсивність навантаження на CS1	3210 викликів/год
Інтенсивність навантаження на CS2 и CS3	3100 викликів/год
Інтенсивність навантаження згідно проєктної потужності	3157 викликів/год
Політика переадресації	Адаптивний режим 2 (Fixed sending/receiving)

Продовження таблиці 4.1

1	2
Тривалість подій виклику	Усічена експонента (середнє значення 5 секунд)
Кількість PE	3
Політика обміну даними	Віддалений доступ
Чинник локальності	50 %
Поріг політики перенесення	175 мс
Поріг політики міграції	105 мс

Параметри робочого навантаження змінюються для дослідження впливу параметрів на поведінку кластера Call Server. Важливими параметрами робочого навантаження є інтенсивність навантаження, час очікування, час обслуговування та фактор локальності. Значення параметрів було вибрано для моделювання кластера Call Server, який охоплює велику географічну область, та комутує виклики, що передають голосовий трафік та трафік даних між абонентами.

Дослідження проведено з використанням розміру кластера на три сервери. Такий розмір кластера може забезпечити комутацію викликів у великій географічній області, наприклад, у всьому місті. Очікується, що відносна продуктивність політик переадресації буде однаковою для кластерів на чотири сервери.

Інтенсивність навантаження - це середня кількість викликів, що надійшли в одиницю часу. Передбачається, що процес надходження викликів є процесом Пуасона. Інтенсивність навантаження, що відповідає проєктній потужності моделі Call Server, становить 3157 викликів/год.

Час очікування включає: час, що витрачається абонентом на набір номера, відповідь на виклик, розмову та відключення після того, як абонент покладе слухавку. Тривалість подій часу очікування визначається зрізаним експоненційним розподілом (0,5-10 с) із середнім значенням 5 с. Експоненційний розподіл призводить до зміни тривалості подій часу очікування, а усічення робиться для виключення дуже високих або дуже низьких значень (що менші, ніж 0,5 с) часу очікування, які неможливі у реальних сценаріях.

Час обслуговування – це час, необхідний для обробки різних повідомлень для обробки викликів. Час обслуговування різний для різних повідомлень.

Дослідження проводиться для високошвидкісної мережі. Передбачається, що швидкість з'єднання між двома серверами обробки викликів становить 100 Мбіт/с. Через невелику довжину повідомлень, що передаються таким виділеним каналом, передбачається, що затримка взаємодії між периферійним модулем і сервером обробки викликів незначна в порівнянні з часом обробки виклику.

Фактор локальності - це ймовірність того, що дані про стан абонента, що викликається, доступні локально на сервері викликів, призначеному для обробки виклику при його надходженні. Передбачуване значення за замовчуванням для коефіцієнта локальності становить 50 %. Фактор локальності 50 % означає, що ймовірність того, що дані будуть доступні локально, дорівнює ймовірності того, що дані будуть розташовані на віддаленому сервері обробки викликів.

4.3 Показники продуктивності кластера серверів

У роботі розглянуто низку показників продуктивності для оцінки відносної ефективності різних політик переадресації.

Затримка гудка (Dial Tone Delay) – це затримка між підняттям трубки телефону та прослуховуванням гудка. Розраховується як різниця між часом, коли периферійний модуль отримує повідомлення "Початок збору даних" від сервера викликів, і часом, коли нове повідомлення "Джерело" надсилається периферійним модулем на сервер виклику. Розраховуються як за допомогою , так і середнього значення затримки гудка.

Затримка походження (Origination Delay) – ключовий вихідний параметр та основний показник продуктивності Call Server. Ця затримка є різницею між часом, коли повідомлення "Джерело" обслуговується підсистемою обробки викликів, і часом, коли процес введення/виведення на сервері викликів отримує це повідомлення "Джерело". Розраховується, як за допомогою , так і середньої затримки відправлення. Ці параметри розраховуються для кожного Call Server у кластері.

Середня затримка встановлення (Average Progress Delay) - середній час виконання: це середній час, що витрачається повідомленнями про перебіг від процесу введення/виведення до підсистеми обробки викликів. Всі повідомлення, окрім повідомлення "Джерело", є повідомленнями про хід виконання. Приклади повідомлень про хід виконання включають повідомлення "Дані", "Відповідь" та "Вихід".

Середній час встановлення з'єднання (Average Setup Delay) – середній час, необхідний для встановлення з'єднання. Ця затримка вимірюється як різниця між часом, коли периферійний модуль починає чути звуковий сигнал виклику телефону адресата, та часом, коли повідомлення "Дані" відправляється з периферійного модуля на сервер обробки викликів.

Середня тривалість процесу (Average Process Length) - середній час, що витрачається на процес щоразу, коли він викликається планувальником. Середня довжина процесу або час процесу вимірюється окремо для процесів введення/виведення, брокера та обробки викликів.

Середнє завантаження ЦП (Average CPU Utilization) – це середнє завантаження ЦП процесом. Він вимірюється як відсоток від загального часу обробки процесу, що витрачається будь-який доступний елемент обробки. Він розраховується окремо для процесів введення/виведення, брокера та обробки викликів.

Середній час між викликами (Average Inter-Invocation Time) – середній час очікування між двома послідовними викликами процесу. Ця метрика обчислюється для процесів введення/виведення та брокера.

Загальна 95-процентільна затримка вихідного виклику (Overall 95th Percentile Origination Delay) – загальна сукупна 95-процентільна затримка вихідного виклику кластера серверів виклику розраховується на основі даних, зібраних для всіх окремих серверів виклику. Це 95-й процентіль затримки вихідного виклику для всіх викликів, оброблених набором із серверів виклиів у кластері.

Чинник поліпшення затримки відправлення ODI (Origination Delay Improvement) - ефективність методів розподілу навантаження вимірюється з погляду, наскільки зменшується затримка виникнення P_{POD} в порівнянні зі значенням, що спостерігається без використання методів розподілу навантаження. Таким чином, визначається поліпшення P_{POD} затримки як відношення без розподілу навантаження до P_{POD} з розподілом навантаження. Це називається фактором ODI. Таким чином, для заданої стратегії розподілу навантаження фактор ODI характеризується виразом:

$$\mathrm{ODI} = \frac{P_{POD(NO-LS)}}{P_{POD(LS)}}, \qquad (4.1)$$

де $P_{POD(NO-LS)}$ − це загальна затримка P_{POD} кластера серверів викликів без розподілу навантаження, а $P_{POD(LS)}$ − це загальна затримка P_{POD} кластера серверів викликів з розподілом навантаження.

ODI повинен бути більшим за 1 для кращого балансування навантаження системи. Чим вище значення фактора ODI, тим кращим є розподіл навантаження між різними серверами викликів кластера.

Основна увага у дослідженнях приділяється вивченню ефективності стратегій розподілу навантаження. Усі стратегії розподілу навантаження відрізняються лише політиками переадресації.

4.4 Дослідження впливу порога політики перенесення та порога політики міграції на продуктивність системи

Існує два типи порогів, що відповідають двом різним компонентам розподілу навантаження. Один поріг відноситься до політики перенесення, а інший – до політики міграції. Ці пороги відповідають кордону між перевантаженою та неперевантаженою системами. Ця гранична лінія є проєктною потужністю системи, але P_{POD} в цій граничній лінії позначається T_E. Сервер обробки викликів, у якого більше P_{POD}, ніж T_E, вважається перевантаженим, а той, у якого P_{POD} нижче T_E, вважається працюючим як слабозавантажений CS.

У першому випадку поріг політики міграції зафіксовано на рівні 140 мс (T_E) та P_{POD} відповідає проєктній потужності CS (E), а поріг політики перенесення зафіксовано в межах від 100 до 240 мс. Для оцінки ефективності розподілу навантаження, інтенсивність навантаження CS1 підтримується трохи вище E (2,5 %), а CS2 та CS3 працюють на 2,5 % нижче E. За допомогою одержаних результатів продуктивності системи для різних значень порога політики перенесення побудовано діаграми, які показано на рис. 4.4.

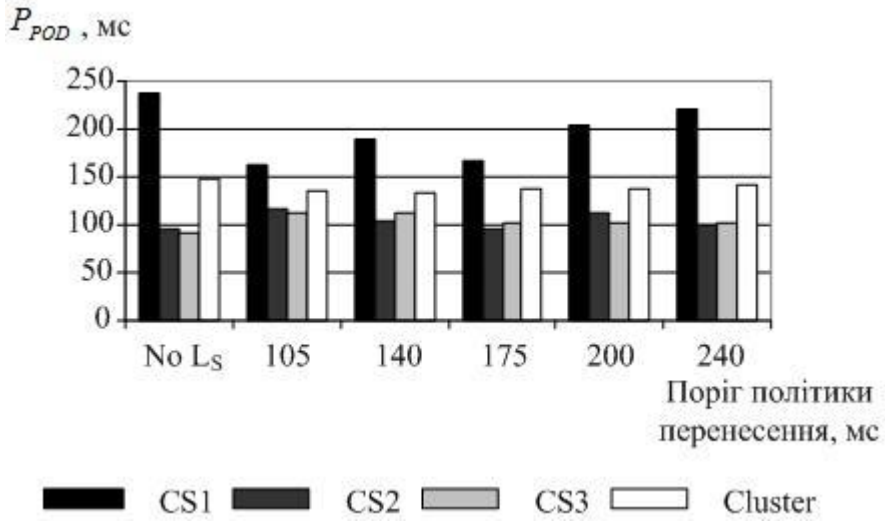

Рисунок 4.4 - Результати продуктивності системи при різних значеннях порога політики перенесення

Діаграма на рис. 4.4 відображає показники, коли метод розподілу навантаження не використовується (No L_S) та показники, отримані при використанні розподілу навантаження. Причому у другому випадку спостерігається покращення продуктивності системи. На діаграмі показано: сервер CS1, який перевантажений, сервери CS2, CS3 і кластер. Чим ближче значення P_{POD} до порогового, тим краще досягається розподіл навантаження у системі.

При пороговому значенні 105 мс P_{POD} перевантаженого CS1 значно знижується, але також підштовхує P_{POD} слабонавантажених CS2 та CS3 до вищих значень. Таким чином, загальний P_{POD} кластеру трохи вищий в порівнянні з результатами із іншими значеннями порога. Аналогічна поведінка спостерігається й у показниках порогу 140 мс.

При пороговому значенні 175 мс P_{POD} CS1 значно знижується без збільшення P_{POD} слабонавантажених CS до вищих значень. Таким чином, загальний P_{POD} кластера кращий за нижні порогові значення (105 мс й 140 мс). Це пов'язано з тим, що перевантажений CS уникає переведення викликів через невеликі сплески навантаження на CS. Для більш високих значень порога, таких як 200 мс та 240 мс, P_{POD} перевантаженого CS1 не сильно зменшується, і тому загальний P_{POD} кластера також високий.

Таким чином, загальний P_{POD} кластера кращий для порогового значення 175 мс. Надалі використано граничне значення 175 мс через його несприйнятливість до невеликих та тимчасових сплесків навантаження.

В роботі проведено дослідження впливу порога політики міграції на загальну продуктивність кластера Call Server. Порогове значення політики міграції використовується для визначення потенційних серверів-отримувачів викликів. У цьому випадку зберігається значення порогу для політики перенесення рівним 175 мс і змінюється значення порога для політики міграції від 70 мс до 175 мс. На рис. 4.5 показано діаграми, що побудовано за результатами дослідження P_{POD} для окремих CS та кластеру.

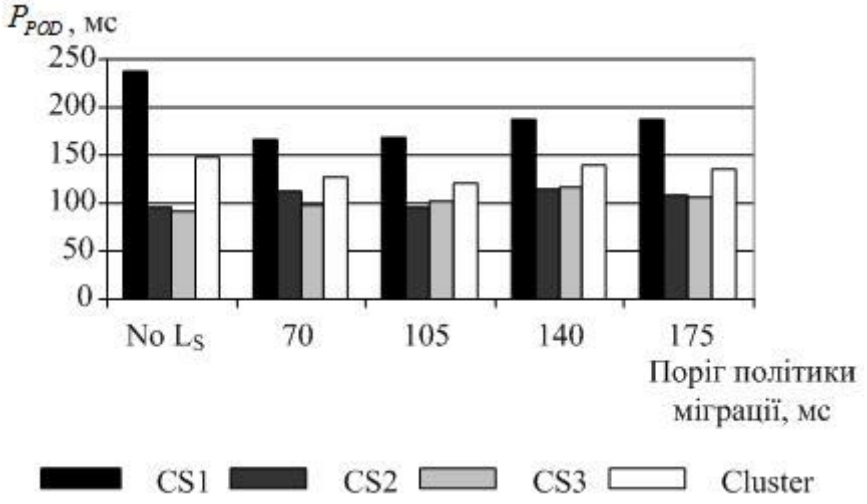

Рисунок 4.5 - Результати продуктивності системи для різних значень порога політики міграції

Видно, що результати для порогових значень 70 і 105 мс кращі, ніж результати, що отримано з пороговими значеннями 140 і 175 мс. Це пов'язано з більш чутливою поведінкою Call Server, коли він працює поблизу значення граничної потужності. Невелике збільшення нових вхідних викликів від віддаленого CS-відправника може привести до перевантаження CS-отримувача. Таким чином, поріг політики міграції має бути обрано нижче P_{POD}, що відповідає проєктній потужності CS.

Загальні P_{POD} для порогових значень 70 і 105 мс знаходяться в межах ± 5 % один від одного.

Таким чином, показано, що в даному дослідженні вибирається 105 мс як поріг політики міграції через те, що це підвищить ймовірність знаходження

приймаючих CS в кластері Call Server в порівнянні з ситуацією, коли у якості порогу використовується значення 70 мс.

4.5 Дослідження режимів переадресації

У роботі проведено дослідження ефективності різних режимів переадресації. Проведено аналіз для різних значень ключових атрибутів. Аналіз проведено на основі навантаження відправника та відправника/одержувача для різних режимів переадресації. Результати цих досліджень дають уявлення про продуктивність системи. Відзначено, що один CS1 перевантажений на 2 % вище E, а інші CS2 та CS3 працюють на 2 % нижче E.

4.5.1 Аналіз режиму переадресації на основі завантаження відправника

У цьому дослідженні на основі завантаження відправника коефіцієнт перенаправлення δ є адаптивним (адаптивний режим 1 - Ending CS) та змінюється залежно від поточного стану завантаження відправляючого CS. Чим вище навантаження на CS-відправник, тим більше кількість викликів, переданих на CS-отримувач. У цьому випадку P є константою, яка відповідає максимальній частці переадресованих вхідних викликів. Значення P може вибиратися від 0 до 100 %. Проведено аналіз впливу різних значень P на продуктивність системи. Оцінка проводиться зі значення 25 %, а потім збільшується з кроком 15 %. Діаграму результатів аналізу показано на рис. 4.6.

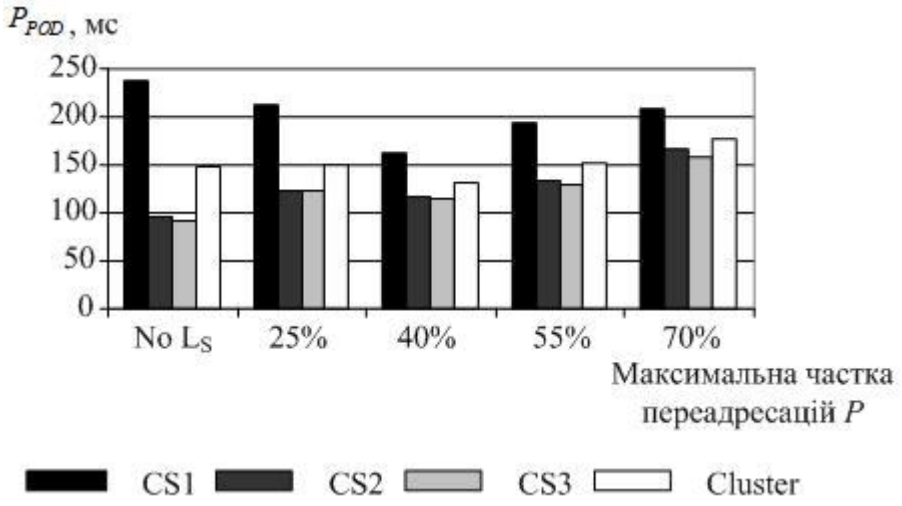

Рисунок 4.6 - Результати впливу P на продуктивність кластеру Call Server

При використанні політики режиму переадресації на основі завантаження відправника як методу розподілу навантаження найкращі результати спостерігаються при P = 40 %. При всіх інших значеннях P показник P_{POD} перевантаженого CS1 збільшується, кластер деградує в порівнянні з загальним P_{POD} без використання будь-якої техніки розподілу навантаження.

Для P = 25 % кількість переадресованих викликів з перевантаженого CS1 на слабонавантажені CS2 та CS3 недостатньо велике. Таким чином, P_{POD} перевантаженого CS1 покращується лише незначно, а P_{POD} слабонавантажених CS2 та CS3 також не сильно збільшується. Загальний P_{POD} кластера дещо гірший у порівнянні з ситуацією без використання розподілу навантаження.

Для P = 40 % як P_{POD} перевантаженого CS1, так і загальний P_{POD} кластера істотно покращуються в порівнянні з ситуацією без використання методу розподілу навантаження.

При більш високих значеннях P, таких як 55 % та 70 %, спостерігається погіршення характеристик. Слабонавантажені CS2 та CS3 показують досить високі значення, але й P_{POD} перевантаженого CS1 відповідно не зменшується. Перевантажений сервер викликів CS1 намагається надіслати занадто багато викликів на приймаючі CS2 та CS3. Це перевантажує сервери викликів-отримувачів, які починають повертати частину своїх викликів на спочатку перевантажений сервер викликів CS1. Цей факт впливає на продуктивність системи двома різними способами. По-перше, це може призвести до переведення нових викликів із серверів викликів, які спочатку були слабозавантажені. По-друге, це ускладнює передачу нових викликів із спочатку перевантаженого сервера викликів, тому що кількість потенційних серверів-отримувачів тепер зменшена.

Таким чином, спостерігаються кращі результати значень P_{POD} для P = 40 %. Тому для решти експериментів використовується P = 40 % для режиму переадресації з урахуванням завантаження відправника.

4.5.2 Аналіз режиму переадресації на основі завантаження відправника/одержувача

Даний режим (адаптивний режим 2 - Fixed sending/receiving CS) враховує стан завантаження як відправляючого, так і приймаючого CS для розрахунку коефіцієнта переадресації вхідних викликів під час виконання. Для цього режиму приймається набір із чотирьох значень (P_1, P_2, P_3, P_4) у порядку зростання, причому кожне значення P відповідає одному рівню доступності CS-приймача. Значення P для дослідження кластера Call Server показані у таблиці 4.2.

Таблиця 4.2 - Значення P для дослідження кластера Call Server

Мітки	Значення (P_1, P_2, P_3, P_4)
P_{11}	(6,12,18,25)
P_{22}	(10,20,30,40)
P_{33}	(14,28,42,55)
P_{44}	(18,36,54,70)
P_{55}	(25,50,75,100)

На основі рівня готовності n приймача CS вибирається відповідне значення P. Для розрахунку використовується вираз (3.6). У цьому дослідженні оцінюється вплив P на продуктивність кластера Call Server (табл. 4.2). Діаграми, побудовані за наслідками дослідження, показано на рис. 4.7.

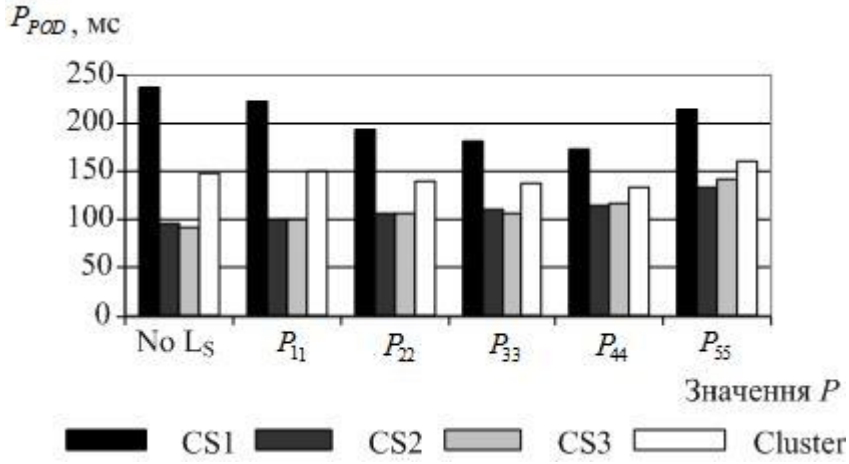

Рисунок 4.7 - Вплив наборів P на продуктивність кластера Call Server

Аналіз рис. 4.7 показав, що результати для набору P_{11} незадовільні через меншу передачу нових викликів з перевантаженого сервера викликів CS1 на слабозавантажені сервери викликів CS2 і CS3. Показник P_{POD} CS1 лише трохи покращено, а вплив P_{POD} на слабонавантажені CS також незначний, як показано в результатах. Загальний P_{POD} для комплекту P_{11} також вказує на відсутність поліпшення продуктивності.

У міру збільшення значень у наборах P спостерігається більше покращення продуктивності кластера Call Server. Для набору P_{22} показник P_{POD} перевантаженого CS1 скорочується до 194 мс, а загальний P_{POD} складає 139 мс. У наборі P_{33} використовуються вищі значення (14,28,42,55 мс). Показник P_{POD} перевантаженого CS та всього кластера додатково покращуються до 181 мс та 137 мс відповідно. Поліпшення продовжується з набором P_{44} (18,36,54,70 мс). Але ця тенденція не продовжується з P_{55}, який включає найвищий рівень частки переадресації (100 %). Результати, отримані для P_{55} показують, що такі високі значення погіршують продуктивність кластера.

Насправді, велика передача нових викликів із перевантаженого CS1 призводить до перевантаження слабозавантажених серверів викликів CS2 та CS3. Це видно за вищими значеннями P_{POD} слабонавантажених CS (132 мс та 142 мс відповідно). А ось P_{POD} CS1, спочатку перевантажений, лише скоротився з 238 мс (без навантаження) до 214 мс. Це можливо через переадресацію викликів назад на CS1 з CS2 та CS3 у ситуації, коли CS1 став слабозавантаженим протягом невеликого проміжку часу. Таким чином, для подальшого аналізу вибирається набір P_{44}.

4.6 Дослідження ефективності політик переадресації з використанням різних рівнів навантаження

В роботі проведено дослідження, що полягають у вивченні ефективності різних політик переадресації для різних конфігурацій незбалансованого навантаження у кластері Call Server. У цьому контексті представлено дві категорії досліджень:

1) Дослідження ефективності всіх політик переадресації, коли інтенсивність навантаження одного перевантаженого CS змінюється вище E на невеликі значення (0-7,5 %) - варіант з одним перевантаженим сервером обробки викликів.

2) Проведення дослідження для визначення ефективності всіх режимів переадресації для дуже високих рівнів навантаження, таких як 25 % і 30 % вище E .

4.6.1 Аналіз кластера Call Server з одним перевантаженим сервером викликів

Умовами для аналізу є: інтенсивність навантаження сервера викликів CS1, що змінюється від 0 % до 7,5 % вище E ; інтенсивність навантаження двох серверів обробки викликів CS2 і CS3 з підтримкою на 5% нижче за розрахункову потужність.

Завантаження конфігурацій для сценарію з одним перевантаженим сервером викликів показано в таблиці 4.3. Представлено різні конфігурації навантаження з одним перевантаженим та двома слабозавантаженими серверами обробки викликів.

Таблиця 4.3 - Завантаження конфігурацій для сценарію з одним перевантаженим сервером викликів

Конфігурації Call Server	Інтенсивність навантаження (викликів /год)	
	Перевантажений CS1	Слабозавантажені CS2 та CS3
1П-2С-1 Ending CS	3150 (E)	3000
1П-2С-2 Fixed sending/receiving	3220 (на 2,5 % вище E)	3000
1П-2С-3 Receiving CS	3300 (на 5 % вище E)	3000
1П-2С-4 Variable sending/receiving CS	3150 (на 7 % вище E)	3000

Діаграму для аналізу загального P_{POD} кластера щодо різних змін навантаження показано на рис. 4.8.

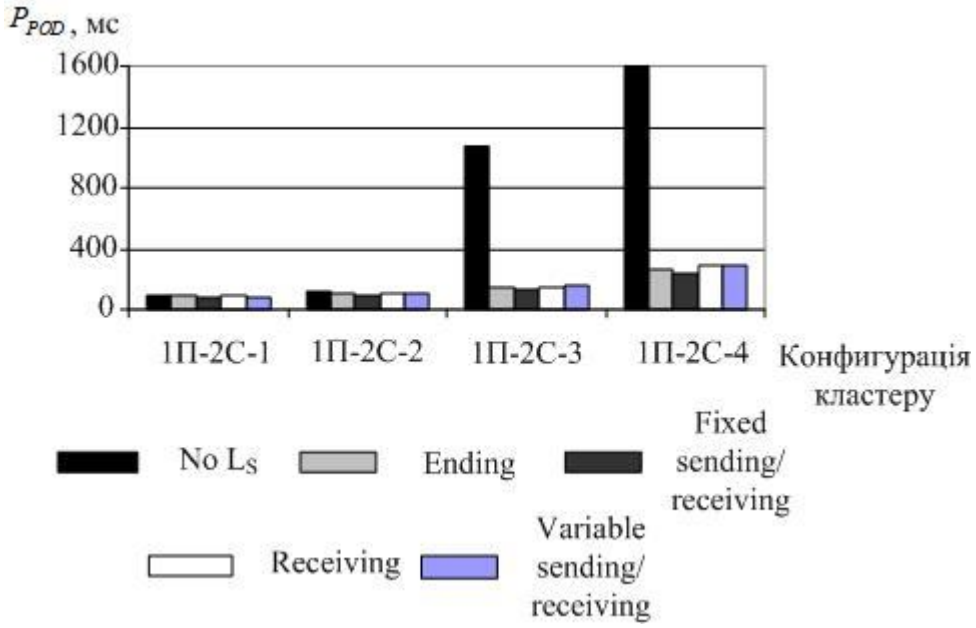

Рисунок 4.8 - Результати P_{POD} кластера для різних конфігурацій навантаження

Режим Ending CS досить добре працює для більшості конфігурацій навантаження. Параметр P_{POD} для останньої конфігурації навантаження (1П-2С-4) досить високий. Режим Fixed sending/receiving CS також добре працює для всіх конфігурацій навантаження. Це пов'язано з його здатністю адаптувати значення δ у відповідності зі станом завантаження як відправляючого, так і отримуючого CS. Таким чином, для цього діапазону конфігурацій навантаження режим переадресації на основі завантаження відправника/одержувача показує кращу продуктивність в порівнянні з іншими режимами.

Режими Receiving CS і Variable sending/receiving CS показують задовільну продуктивність, окрім останньої конфігурації навантаження (1П-2С-4). Обидва режими показують незначні поліпшення в P_{POD} для цієї конфігурації. Це пов'язано з тим, що обидва режими активніше перенаправляють викликі, ніж інші режими. Перевантажений CS працює на 7,5 % вище E, а слабонавантажені CS працюють лише на 5 % нижче E. Таким чином, при переадресації викликів з більшою швидкістю ці режими зрештою перевантажують одержувача.

Однак на рис. 4.8 відображено P_{POD} кластера без використання будь-якої техніки розподілу навантаження для кожної конфігурації (No L_S). Таким чином, аналіз показав, наскільки ефективно режими переадресації зменшують P_{POD} кластера в порівнянні з P_{POD} без розподілу навантаження.

В роботі проведено оцінку кластера Call Server з допомогою коефіцієнта ODI. Діаграми за результатами аналізу коефіцієнта ODI при різних конфігураціях навантаження показано на рис. 4.9.

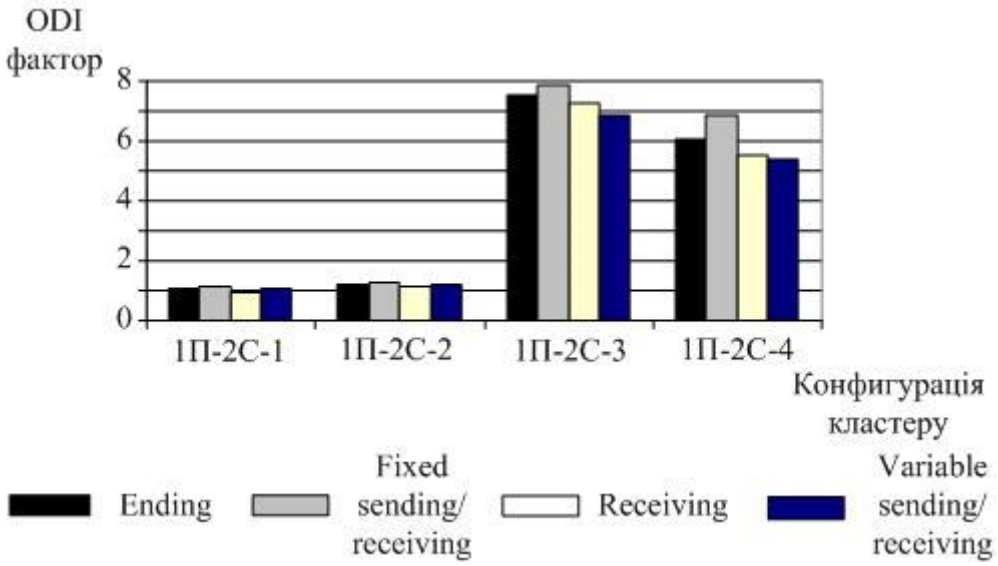

Рисунок 4.9 - Результати аналізу коефіцієнта ODI при різних конфігураціях навантаження

Найкращі результати коефіцієнта ODI при різних конфігураціях навантаження досягнуто за допомогою режиму Fixed sending/receiving CS.

У роботі проведено оцінку середньої затримки вихідного трафіку всіх режимів переадресації.

Середню затримку ініціації (вихідного виклику) при різних конфігураціях навантаження без розподілу навантаження для варіанта з одним перевантаженим сервером викликів показано на рис. 4.10.

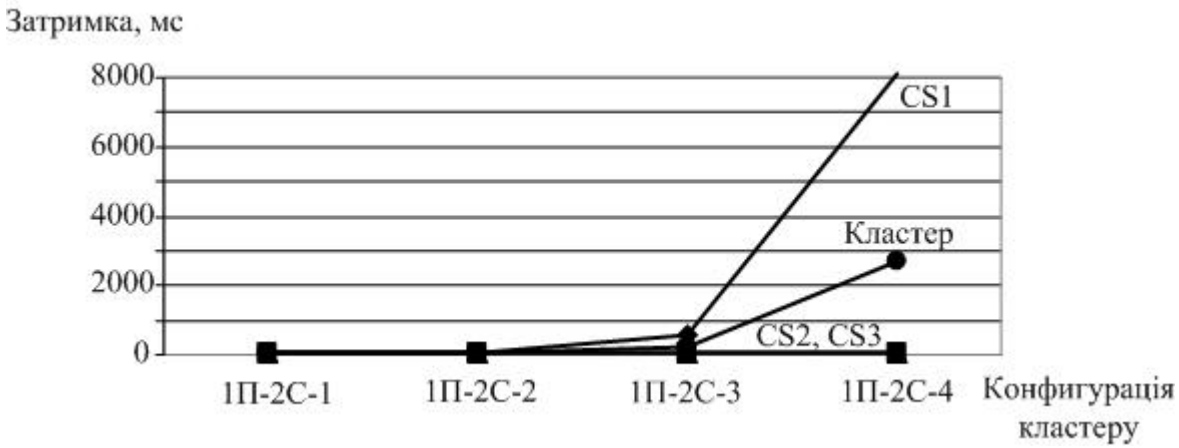

Рисунок 4.10 - Середня затримка ініціації для варіанта з одним перевантаженим сервером викликів

Середню затримку ініціації, яку отримано при різних конфігураціях навантаження в режимі Ending CS для варіанта з одним перевантаженим сервером викликів, показано на рис. 4.11.

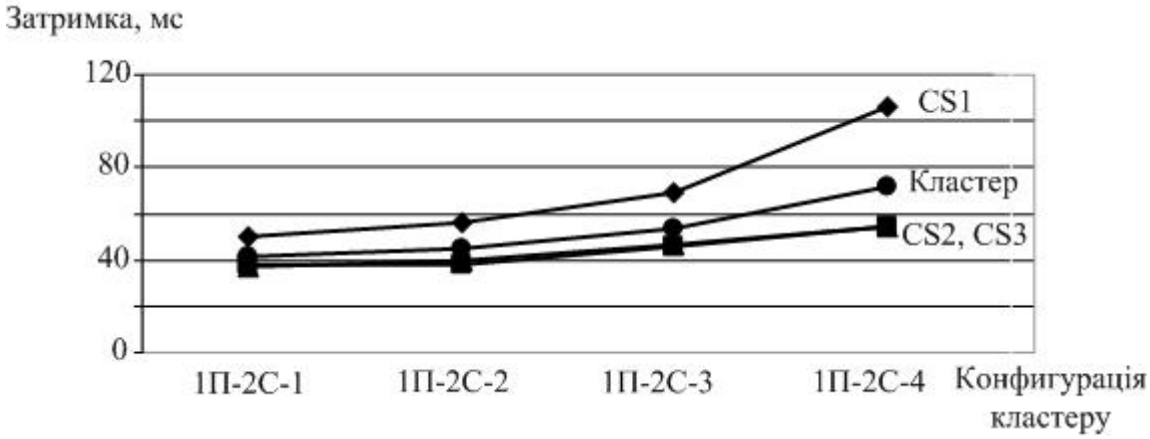

Рисунок 4.11 - Середня затримка ініціації при різних конфігураціях навантаження в режимі Ending CS

Середню затримку ініціації, яку отримано при різних конфігураціях навантаження в режимі Fixed sending/receiving CS для варіанта з одним перевантаженим сервером викликів, показано на рис. 4.12.

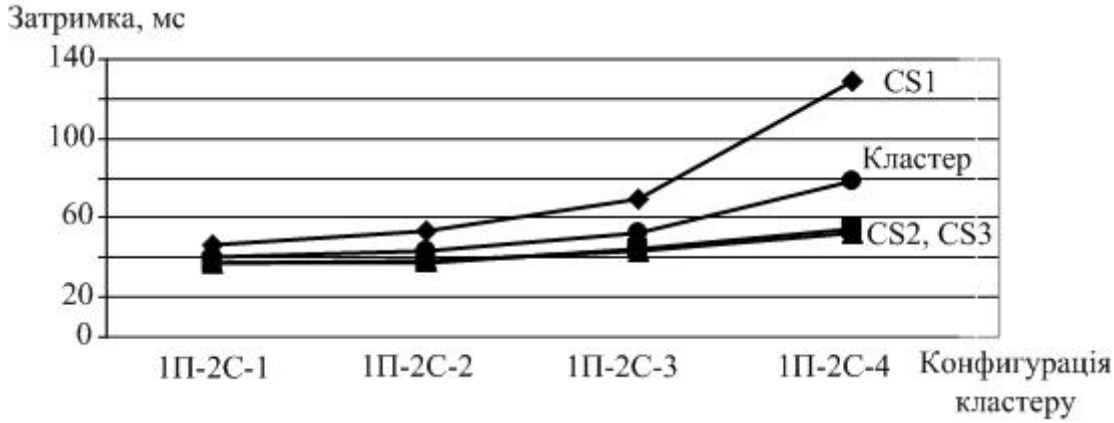

Рисунок 4.12 - Середня затримка ініціації при різних конфігураціях навантаження в режимі Fixed sending/receiving CS

Середню затримку ініціації, яку отримано при різних конфігураціях навантаження в режимі Receiving CS для варіанта з одним перевантаженим сервером викликів показано на рис. 4.13.

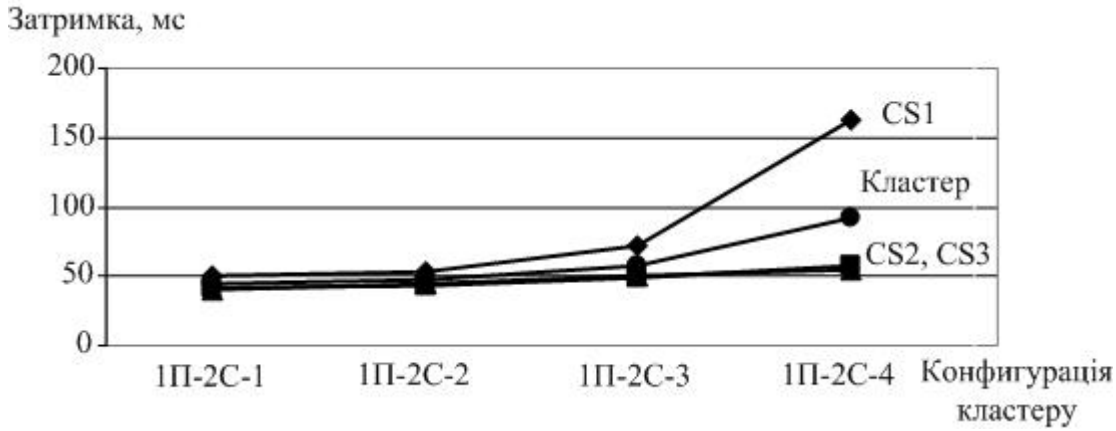

Рисунок 4.13 – Середня затримка ініціації при різних конфігураціях навантаження у режимі Receiving CS

Середню затримку ініціації, яку отримано при різних конфігураціях навантаження в режимі Variable sending/receiving CS для варіанта з одним перевантаженим сервером викликів показано на рис. 4.14.

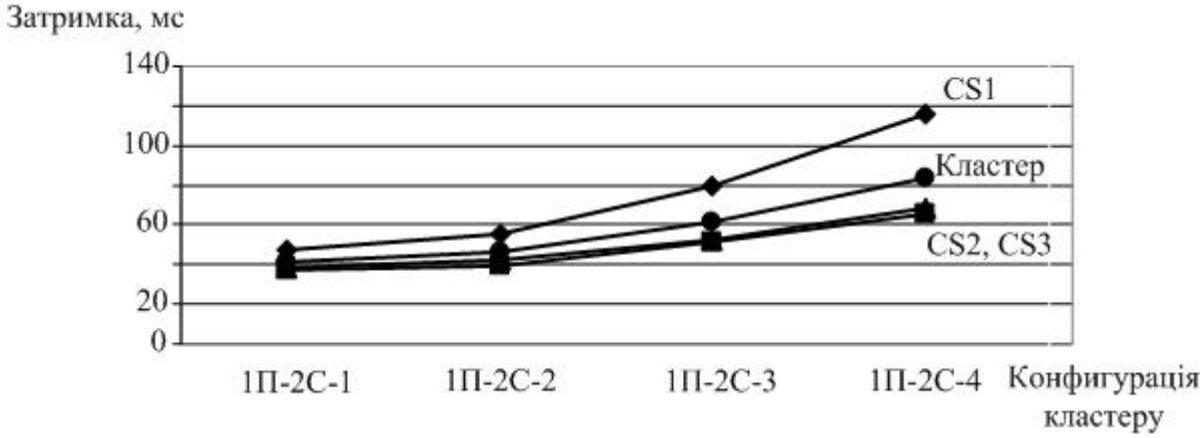

Рисунок 4.14 - Середня затримка ініціації при різних конфігураціях навантаження в режимі Variable sending/receiving CS

Таким чином, найбільш прийнятні результати, що перебувають у менших значеннях середньої затримки ініціації кластера, можна спостерігати при конфігурації навантаження в режимі Variable sending/receiving CS. Значення затримки при конфігурації 1П-2С-1 становлять 70 мс.

4.6.2 Аналіз кластера Call Server при високих рівнях навантаження

З метою аналізу ефективності режимів переадресації при більш високих рівнях перевантаження проведено дослідження, в яких передбачається, що один сервер обробки викликів CS1 перевантажений на 25 % і на 30 % вище E, а два слабозавантажені сервери обробки викликів CS2 і CS3 працюють на 50 % нижче E. Проведено оцінку двох різних рівнів навантаження з метою виявлення більш стабільної продуктивності різних режимів переадресації. Діаграми, побудовані за результатами аналізу для зазначених рівнів навантаження CS, представлено на рис. 4.15.

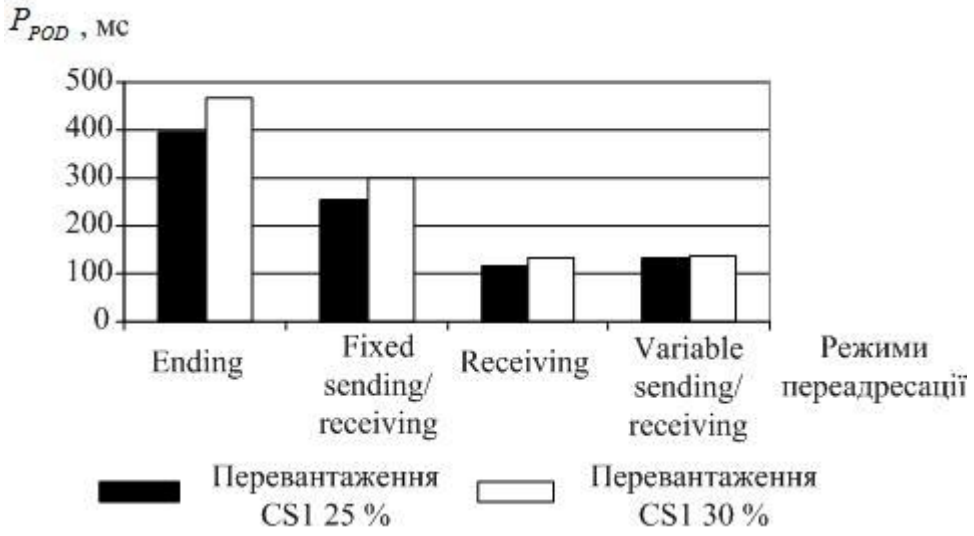

Рисунок 4.15 - Результати аналізу кластера з використанням різних режимів переадресації та заданих рівнях перевантаження

Аналіз показав, що показник P_{POD} для режимів Ending CS і Fixed sending/receiving CS вищі за необхідне значення 140 мс. Однак, режим Fixed sending/receiving CS працює краще, ніж режим Ending CS.

Режими Receiving CS та Variable sending/receiving CS демонструють хорошу продуктивність в порівнянні з іншими режимами переадресації. Їх результати задовольняють необхідному значенню до 140 мс та узгоджуються на обох рівнях (25 % та 30 %) навантаження. Однак, режим Receiving CS показує стабільно найкращий результат на обох рівнях навантаження.

4.7 Дослідження впливу зміни кластера

У роботі проведено дослідження продуктивності кластерної системи для різних розмірів та складів кластера. Для цього аналізу використано три різні конфігурації:

1) 2CS-1П-1С - у цій конфігурації два Call Server у кластері. Один сервер обробки викликів CS1 працює з інтенсивністю навантаження на 5 % вище E, а другий сервер обробки викликів CS2 працює з інтенсивністю навантаження на 5 % нижче E.

2) 3CS-1П-2С - ця конфігурація складається з трьох серверів викликів. Один CS1 перевантажений на 5 % вище E, а два інших CS2 та CS3 працюють на 5 % нижче E.

3) 3CS-2П-1С - ця конфігурація складається з трьох серверів викликів. Два сервери CS1 і CS2 перевантажено на 5 % вище за E, і один CS3 працює на 5 % нижче E.

Результати для трьох конфігурацій кластера без розподілу навантаження показано на діаграмі (рис. 4.16).

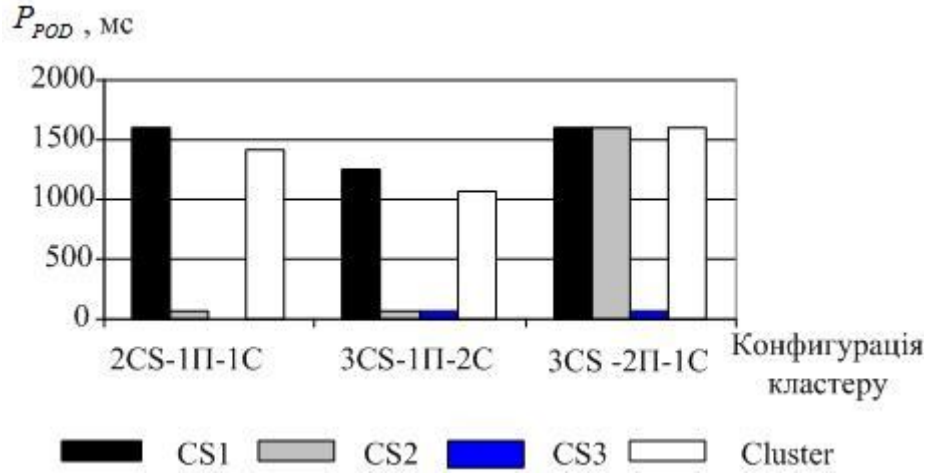

Рисунок 4.16 - Вплив складу кластера Call Server на P_{POD} без розподілу навантаження

Аналіз показав, що для конфігурацій навантаження 2CS-1П-1С й 3CS-2П-1С P_{POD} перевантаженого CS1 і кластерної системи дуже високі в порівнянні з результатами, досягнутими для конфігурації 3CS-1П-2С. Це пов`язано з меншою кількістю слабонавантажених CS для даних конфігурацій кластеру.

Крім власної інтенсивності навантаження, на параметр P_{POD} CS також впливає кількість віддалених серверів обробки викликів й інтенсивність навантаження на ці віддалені сервери обробки викликів. Вища інтенсивність навантаження та більша кількість віддалених CS призводять до більшої кількості запитів обміну даними до локального Call Server. Це є причиною погіршення результатів для конфігурацій 2CS-1П-1С та 3CS-2П-1С.

Результати трьох конфігурацій кластера з розподілом навантаження показано на діаграмі (рис. 4.17).

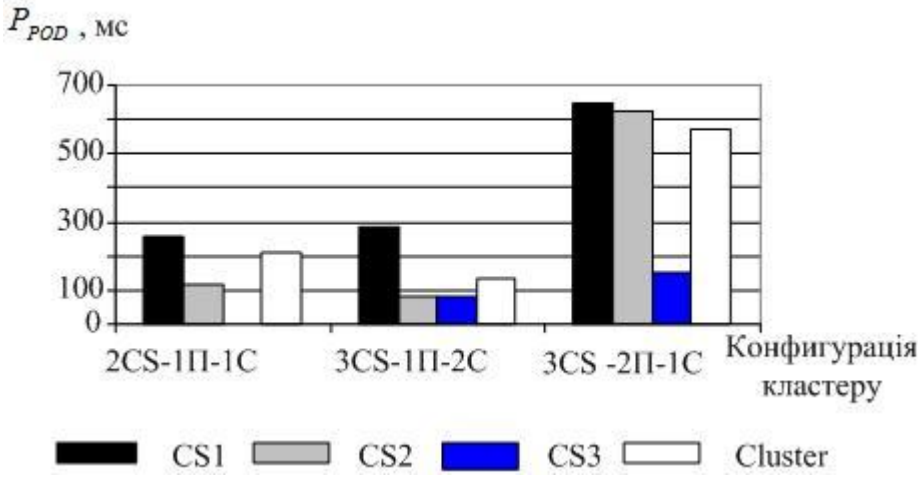

Рисунок 4.17 - Вплив складу кластеру Call Server на P_{POD} з розподілом навантаження

Аналіз рис. 4.17 показав, що розподілення навантаження покращило продуктивність системи для всіх конфігурацій кластера. Але поліпшення продуктивності спостерігається більше для тих конфігурацій, які мають більш високе співвідношення кількості слабозавантажених серверів викликів до перевантаженим серверам викликов. Конфігурація кластера 3CS-1П-2C має два слабонавантажених CS, тому спостерігається найкраще значення загального P_{POD} для цієї конфігурації.

Конфігурація 2CS-1П-1C з одним слабонавантаженим CS і одним перевантаженим CS також показала кращу продуктивність в порівнянні з кластерною конфігурацією 3CS-2П-1C, в якій кількість перевантажених CS більша, ніж кількість слабонавантажених CS.

Результати, які отримано для конфігурації 3CS-2П-1C, показали, що дана методика розподілу навантаження не намагається перевантажити слабонавантажений CS3. Це спостерігається за значенням P_{POD} CS3, тобто помічено збільшення цього значення лише до 150 мс, хоча перевантажені CS1 і CS2 продовжують працювати в перевантаженому стані.

4.8 Дослідження тривалості подій виклику

У роботі розглянуто чотири події виклику, які залежать від відповіді абонентів: набір номера, відповідь на дзвінок, розмова та завершення розмови.

Тривалість цих подій виклику величина змінна. У роботі проведено моделювання цих подій за допомогою зрізаних експоненційних розподілів, тривалість яких змінюється від секунд до хвилин залежно від виклику. Результати впливу різної тривалості подій виклику на P_{POD} кластера Call Server з розподілом навантаження та без розподілу навантаження показано на діаграмі (рис. 4.18).

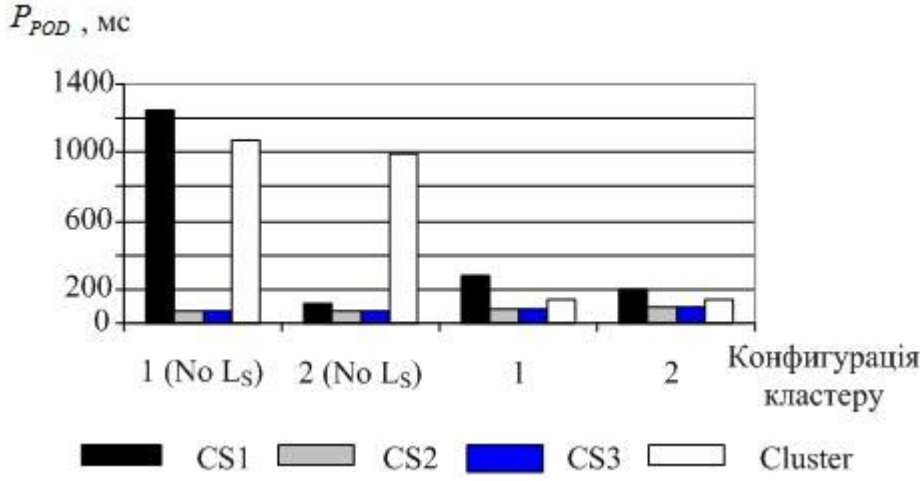

Рисунок 4.18 - Результати впливу різної тривалості подій виклику на P_{POD} кластера Call Server

На схемі 1 середнє значення подій виклику вибрано тривалістю 5 с. На схемі 2 для тривалості різних подій виклику використовуються різні середні значення часу: 3 с для набору номера, 5 с для відповіді на виклик, 10 с для розмови та 2 с для завершення виклику.

Різні значення часу на схемі 2 відображають той факт, що набір номера займає менше часу, ніж телефонна розмова. Так само повісити трубку - це найменший час, що вимагає активності. Загальна середня тривалість всіх чотирьох подій на схемі 2 становить 5 с, що дорівнює загальній середній тривалості чотирьох подій на схемі 1.

Таким чином, один із серверів обробки викликів CS1 працює з інтенсивністю навантаження, що дорівнює 5 % вище E, а два інших Call Server CS2 та CS3 працюють з інтенсивністю навантаження на 5 % нижче E.

На рис. 4.18 також показано результати для двох схем з використанням розподілу навантаження та без розподілу навантаження (No L_S). Без розподілу навантаження схема 2 дає кращі результати, ніж схема 1. Отримані результати відображають той факт, що немає великої різниці в загальному значенні P_{POD}

(±7,84 % без розподілу навантаження та ±0,2 % з розподілом навантаження) у кластері, отриманому з допомогою схеми 1 та схеми 2, особливо з розподілом навантаження. Результати показують, що продуктивність системи не дуже чутлива до того, чи використовуються ті самі або різні засоби для моделювання подій виклику.

ВИСНОВКИ

В кваліфікаційній роботі проведено дослідження динамічних політик переадресації та характеристик кластерів серверів обробки викликів з метою оцінки їх впливу на продуктивність системи за різних поєднань параметрів робочого навантаження, а також за різних сценаріїв перевантаження.

Проаналізовано особливості функціонування та реалізації мережі Call-центру. У якості ядра даної структури виступає кластер серверів з використанням віртуальної IP-АТС Asterisk. Такий вибір обумовлено модульною архітектурою Asterisk та можливістю працювати на багатьох ОС.

З метою проведення подальших досліджень в роботі розглянуто особливості процесу моделювання телефонної мережі. Доведено, що для огляду деяких аспектів функціонування мережі необхідна віртуалізація всієї системи, або деяких її частин.

Проведено огляд особливостей моніторингу телефонної мережі.

Для розробки та налаштування системи моніторингу в роботі створено мережну модель Call-центру. Мережна модель визначає загальну структуру об'єктів, а також їх фізичне розташування, взаємозв'язки та конфігурації.

Для створення мережі Call-центру та необхідного тестування використано технологію віртуалізації. У якості платформи для налаштування мережі використано гіпервізор VMWare ESXI 6.7 та кліент vCenter.

В роботі проведено налаштування АТС Astersik та створення кластеру серверів. В мережі Call-центру ВМ Asterisk_1 виступає як в ролі АТС, так і в ролі сервера балансування навантаження.

Для дослідження кластеру серверів викликів в роботі розроблено та налаштовано систему моніторингу з використанням open source продукту Zabbix. У роботі запропоновано його застосування для дослідження кластеру серверів викликів, які змодельовано як географічно розподілені сегменти телекомунікаційної мережі. Для візуалізації даних в Zabbix отримано характеристику пропускної здатності для сервера Astersik_1 та карту мережі, яка фактично являє собою змодельовану структуру мережі Call-центру.

В роботі показано процес тестування навантаження викликами на три сервера Asterisk та реалізація можливостей Zabbix як інструменту балансування навантаження. При цьому розрахована кількість викликів, з якими здатний

працювати Call-центр. Виявлено, що один сервер Asterisk з поточними його параметрами здатний обробити максимум 915 одночасних викликів.

В роботі запущено процес балансування навантаження на кластер SIP серверів, виходячи з метрик стану CPU та стану SIP транку. Виявлено, що кластер серверів Asterisk здатний обслуговувати 2550 викликів одночасно.

В роботі проведено огляд літератури щодо питань балансування навантаження з використанням різних алгоритмів та стратегій. Розроблено структурну модель Call Server cluster для кластеру з трьох серверів викликів. Для аналізу продуктивності кластеру серверів викликів обрано характеристику індексу навантаження, у якості якої виступає затримка 95-го процентілю - 95-й POD.

В роботі проведені дослідження, пов’язані як з дослідженням одиночного CS у кластері, так і з дослідженням кластеру серверів викликів.

Проведено дослідження методів та політик розподілу навантаження у кластері серверів. Проведено дослідження відносних характеристик адаптивного підходу для ефективної обробки стрибків навантаження, що виникають у кластері. Дослідження направлено на розрахунок кількості переадресованих викликів та коефіцієнта переадресації у заданих режимах переадресації та використовує інформацію про завантаження відправляючого та/або приймаючого сервера викликів.

По результатам дослідження побудовано характеристики, аналіз яких показав, що режими Fixed sending/receiving CS і Receiving CS демонструють хорошу продуктивність для широкого діапазону параметрів робочого навантаження. Проте режим Receiving CS вважається кращим вибором, оскільки використовує менше інформації про стан системи.

Для аналізу характеристик CS у кластері побудовано модель вузла (клієнт-серверна модель) та модель процессу (діаграма взаємодії UML).

Проведено дослідження впливу порога політики перенесення та порога політики міграції на продуктивність системи при використанні та відсутності розподілу навантаження. Отримано результати щодо роботи кожного сервера у кластері та кластера в цілому.

В роботі проведено дослідження ефективності чотирьох режимів переадресації. Проведено аналіз для різних значень ключових атрибутів. Результати цих досліджень дають уявлення про продуктивність системи.

Проведено дослідження, що полягають у оцінці ефективності різних політик переадресації для різних конфігурацій незбалансованого навантаження у кластері Call Server. У цьому контексті представлено дві категорії досліджень: варіант з одним перевантаженим сервером обробки викликів та варіант для дуже високих рівнів навантаження.

В роботі отримано результати досліджень середньої затримки вихідного трафіку для всіх режимів переадресації та при при різних конфігураціях навантаження. Згідно оцінки визначено найбільш прийнятні результати, які можна спостерігати при конфігурації навантаження в режимі Variable sending/receiving CS. Значення затримки при конфігурації 1П-2С-1 становлять 70 мс.

Проведено дослідження продуктивності кластерної системи для різних розмірів та складів кластера з розподілом та без розподілу навантаження. Проведено моделювання подій виклику за допомогою зрізаних експоненційних розподілів, тривалість яких змінюється від секунд до хвилин залежно від виклику. Результати показали, що продуктивність системи не дуже чутлива до того, чи використовуються ті самі або різні засоби для моделювання подій виклику.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Колтаков О.А., наук. керівник Токар Л.О. Віртуалізація ресурсів підприємства / О.А. Колтаков, Л.О. Токар // Матеріали IV Міжнародної студентської наукової конференції Наука сьогодення: від досліджень до стратегічних рішень, Івано-Франківськ, 2022. – С.178-180.

2. Колтаков О.А., науковий керівник доц. Ковальчук В.К. Дослідження методів оптичної рефлектометрії / О.А. Колтаков, В.К. Ковальчук // Матеріали 24-го Міжнародного молодіжного форуму "Радіоелектроніка і молодь в 21 столітті", Харків: ХНУРЕ, 2020. – С. 68-69.

3. Токар Л.О., Греков І. С., Колтаков О.А. Необхідність забезпечення справедливості ефірного часу для багатошвидкісних мереж IEEE 802.11 Матеріали сьомої Міжнародної науково-технічної конференції "Проблеми електромагнітної сумісності перспективних безпроводових мереж зв'язку EMC-2021", Харків: ХНУРЕ, 2021. – Том 4. – С. 66-68.

4. Voxlink [Електронний ресурс]. – 2011. – Режим доступу до ресурсу: https://voxlink.com/

5. Bryant R., Madsen L., Meggelen J. V. Asterisk The Definitive Guide / R. Bryant., L. Madsen., J. V. Meggelen. – O`Reilly Media, 2013. – 845 p.

6. Davidson J., Peters J., Bhatia M., Kalidindi S., Mukherjee S. Voice over IP Fundamentals / J. Davidson, J. Peters, M. Bhatia, S. Kalidindi, S. Mukherjee. – Cisco Press, 2007. – 392 p.

7. Поповський В. В. Основи теорії телекомунікаційних систем. Підручник / В. В. Поповський. – Харків: ХНУРЕ, 2018. – 368 с.

8. Токар Л. О. Особливості побудови віртуальних АТС / Л. О. Токар // Радіотехніка. Всеукраїнський міжвідомчий науково-технічний сбірник, Харків: ХНУРЕ, 2022. – Вип. 208. – С. 55-64.

9. Баскаков І. В. IP-телефонія у комп'ютерних мережах / І. В. Баскаков, А. В. Пролетарський, С. А. Мельников, Р. А. Федотов. – Інтернет-університет інформаційних технологій, 2020. – 227 с.

10. Materna B. Proactive security for VoIP networks / B. Materna // Information Systems Security, 2006. – vol. 15. – no. 2. – P. 16–21. doi.org/10.1201/1086.1065898X/46051.15.2.20060501/93405.4.

11. Граннеман Скотт Linux. Карманный справочник / Скотт Граннеман. – Sams Publishing, 2019. – 464 с.

12. Nuno Pelayo, Suárez Carla, Suárez Eva A Diagnosis and Hardening Platform for an Asterisk VoIP PBX / Pelayo Nuno, Carla Suárez, Eva Suárez // Security and Communication Networks, Spain, 2020. – Vol. 2020. – P. 1-14. doi.org/10.1155/2020/8853625.

13. Установка сертификатов в Ubuntu [Електронний ресурс]. –2022. – Режим доступу до ресурсу: https://losst.pro/.

14. Uytterhoeven Patrik, Olups Rihards Zabbix 4 Network Monitoring / Patrik Uytterhoeven, Rihards Olups. – Third Edition Packt, 2019. – 798 p.

15. Van Baekel Brian, Liefting Nathan Zabbix 6 IT Infrastructure Monitoring / Brian Van Baekel, Nathan Liefting. – Packt Publishing, 2022. – 506 p.

16. ZABBIX 6.2 Improve your monitoring performance [Електронний ресурс]. –2022. – Режим доступу до ресурсу: https://www.zabbix.com/.

17. Pradana A., Widiasari I., Efendi R. Implementasi Sistem Monitoring Jaringan Menggunakan Zabbix Berbasis SNMP / A. Pradana, I. Widiasari, R. Efendi // Security and Communication Networks, AITI, 2022. – Vol. 19(2). – P. 248-262. doi.org/10.24246/aiti.v19i2.248-262.

18. Clementia1 Andrea, Nataleb Emanuele, Ziccardic Isabella Parallel Load Balancing on constrained client-server topologies / Andrea Clementia1, Emanuele Nataleb, Isabella Ziccardic // Theoretical Computer Science, 2021. – Volume 895. – P. 16-33.

19. Gardner K., Abdul Jaleel J., Wickeham A., Doroudi S. Scalable load balancing in the presence of heterogeneous servers / K. Gardner, J. Abdul Jaleel, A. Wickeham, S. Doroudi // Performance Evaluation, 2020. – P. 102-151. DOI:10.10163.

20. Кузьминых Е.Д. Анализ методов балансировки нагрузки в кластере SIP-серверов / Е.Д. Кузьминых // Проблемы телекоммуникаций, Харьков: ХНУРЕ. – 2014. – № 2 (14). – С. 67-75.

21. Siokis A., Christodoulopoulos K., Pleros N., Varvarigos E. Electro-optic switches based on space switching of multiplexed WDM signals: Blocking vs non-blocking design trade-offs / A. Siokis, K. Christodoulopoulos, N. Pleros, E. Varvarigos // Optical Switching and Networking, 2017. – Volume 25. – P. 40-56. DOI:10.1016/j.osn.2017.

22. Medhi D., Ramasamy K. Routing and Traffic Engineering using MPLS / D. Medhi, K. Ramasamy // Network Routing, 2018. – P. 766-785. DOI:10.1016/b978-0-12-800737-2.

23. Ataie Reza Ehsan, Sayed Entezari-Maleki, Etesami Ehsan, Egger Bernhard, Sousa Leonel, Movagharg Ali Modeling and evaluation of dispatching policies in IaaS cloud data centers using SANs / Ehsan Ataie Reza, Entezari-Maleki Sayed, Ehsan Etesami, Bernhard Egger, Leonel Sousa, Ali Movagharg // Sustainable Computing: Informatics and Systems, 2022. – Volume 33. – P. 88-102. https://doi.org/10.1016/j.suscom.2021.

24. Michailidis P. D., Margaritis K. G. Performance evaluation of load balancing strategies for approximate string matching application on an MPI cluster of heterogeneous workstations / P. D. Michailidis, K. G. Margaritis // Future Generation Computer Systems, 2003. – Volume 19(7). – P. 1075–1104. DOI:10.1016/s0167-739x(03)00109-2.

25. Ali M., Bagchi S. Probabilistic normed load monitoring in large scale distributed systems using mobile agents / M. Ali, S. Bagchi // Future Generation Computer Systems, 2019. – Volume 96. – P. 148-167. DOI:10.1016/j.future.2019.01.053.

26. Muhuri P. K., Biswas S. K. Bayesian optimization algorithm for multi-objective scheduling of time and precedence constrained tasks in heterogeneous multiprocessor systems / P. K. Muhuri, S. K. Biswas // Applied Soft Computing, 2020. – Volume 92. – P. 1-27. DOI:10.1016/j.asoc.2020.106274.

27. Sabir B. E., Youssfi M., Bouattane O., Allali H. Authentication and load balancing scheme based on JSON Token For Multi-Agent Systems / P. K. Muhuri, S. K. Biswas // Procedia Computer Science, 2019. – Volume 148. – P. 562–570. DOI:10.1016/j.asoc.2020.

28. Graveto V., Rosa L., Cruz T., Simoes P. A. Stealth Monitoring Mechanism for Cyber-Physical Systems / V. Graveto, L. Rosa, T. Cruz, P. A. Simoes // International Journal of Critical Infrastructure Protection, 2018. – 53 p. DOI: doi:10.1016/j.ijcip.2018.10.006.

29. Long Q., Lin J., Sun Z. Agent scheduling model for adaptive dynamic load balancing in agent-based distributed simulations / Q. Long, J. Lin, Z. Sun // Simulation Modelling Practice and Theory, 2011. – Volume 19(4). – P. 1021–1034. DOI: 10.1016/j.simpat.2011.01.002.

30. Decentralized Greedy, Braquet Martin, Bakolas Efstathios Auction-based Task Allocation for Multi-Agent Systems / Greedy Decentralized, Martin Braquet, Efstathios Bakolas // IFAC-Papers OnLine, 2021. – Volume 54. – Issue 20. – P. 675-680.

31. Lee G.H., Lee H.K., Cho J.W. A prediction-based adaptive location policy for distributed load balancing / G.H. Lee, H.K. Lee, J.W. Cho // Journal of Systems Architecture, 1996. – Volume 42(1). – P. 1-18. DOI:10.1016/1383-7621(96)00006-9.

32. Srivastava V., Pandey R. S. Machine intelligence approach: To solve load balancing problem with high quality of service performance for multi-controller based Software Defined Network / V. Srivastava, R. S. Pandey // Sustainable Computing: Informatics and Systems, 2021. – Volume 30. – P. 1-11. DOI:10.1016/j.suscom.2021.

33. ESET Endpoint Security. Посібник користувача [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: file:///C:/Users/RedLine/Downloads/eset_endpoint_security_8_ukr.pdf.

34. Кластеризація в умовах глобалізації [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://ucluster.org/universitet/klastery-svit/klastery-globalna-ekonomika/rozdil-1-pidrozdil-2/5.

35. Черкасов Дмитро Основи технології VoIP та IP-телефонії / Дмитро Черкасов // Національний Університет "Києво-Могилянська Академія" Военная свіязь, 2017. – № 2. – С. 98-104.

36. Черняк О.І., Захарченко П.В. Інтелектуальний аналіз даних. Підручник / О.І. Черняк, П.В. Захарченко. – Київ: Знання, 2010. – 837 с.

37. Посібник користувача кластера [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: http://horst-7.bitp.kiev.ua/Cluster-Manual/.

38. Литвинов О.А., Хандецький В.С. Розподілена обробка інформації. Монографія / О.А. Литвинов., В.С. Хандецький. − Дніпро: ТОВ "Баланс-Клуб", 2013.− 314 с.

39. Cisco CallManager System Guide (Release 3.1), Corporate Headquarters, Cisco Systems Inc. [Електронний ресурс]. – 2001. – Режим доступу до ресурсу: https://www.cisco.com/en/US/docs/voice_ip_comm/cucm/admin/3_1_2/ccmsys/ccmsy.html.

40. Muhammad Asif Load Sharing in Call Server Clusters / Asif Muhammad. − Ottawa, Ontario, Canada: B.Sc, 2005. − 138 p.