Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
Охорона (назва факультету)
Кафедра комп'ютерних систем та мереж
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

Магістр
(назва освітнього ступеня)

на тему: Методи і засоби моніторингу та прогнозування несправностей у розподілених комп’ютерних системах

Виконав: студент 6 курсу групи СІм-61
спеціальності
123 «Комп’ютерна інженерія»
(шифр і назва спеціальності (напряму підготовки))

(підпис) Гладовський Н.О. (прізвище та ініціали)

Керівник (підпис) Баран І.О. (прізвище та ініціали)

Нормоконтроль (підпис) Тиш Є.В. (прізвище та ініціали))

Завідувач кафедри (підпис) Осухівська Г.М. (прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

м. Тернопіль – 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних систем та мереж
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
доц. Осухівська Г.М.
(підпис) (прізвище та ініціали)
11 листопада 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня *магістр*
(назва освітнього ступеня)

за спеціальністю *123 Комп'ютерна інженерія*
студенту *Гладовському Назару Олександровичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Методи і засоби моніторингу та прогнозування несправностей у розподілених комп'ютерних системах*

Керівник роботи *Баран Ігор Олегович., к.т.н.,доц.*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом по університету від *« 29 » листопада 2024 року № 4/7-1144*

2. Термін подання студентом роботи *20.12.2024*

3. Вихідні дані до роботи *наукові літературні джерела*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1 Аналіз предметної області. 2 Теоретична частина.
3. Прогнозування аномалій за допомогою алгоритмів машинного навчання.
4 Охорона праці та безпека в надзвичайних ситуаціях

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема, мета, задачі, об'єкт, предмет, новизна дослідження. 2. Актуальність дослідження.
3. Приклад РКС. 4.Порівняльні характеристики деяких систем моніторингу
5. Схема сукупності систем для автоматизації усунення несправностей.
6. Збір та обробка системних журналів syslog. *7.Блок-схема алгоритму побудови дерев шаблонів повідомлень. 8. Результати експериментів*
9. Висновки. Результати проведеного дослідження..

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	*Осухівська Г.М., доцент*		
Безпека в НС			

7. Дата видачі завдання *11.11.2024.*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	*Аналіз існуючих методів та засобів розв'язання науково-технічних проблем, що відповідають тематиці кваліфікаційної роботи.*	*20.11-30.12.24*	*Виконано*
2.	*Обґрунтування актуальності дослідження*	*01.12-05.12.24*	*Виконано*
3.	*Аналіз предмету дослідження та предметної області*	*06.12-09.12.24*	*Виконано*
4.	*Проведення дослідження методів та засобів аналітичного опрацювання даних*	*10.12-12.12.24*	*Виконано*
5.	*Оформлення розділу «Аналіз предметної області»*	*12.12-13.12.24*	*Виконано*
6.	*Оформлення розділу «Теоретична частина»*	*13.12-14.12.24*	*Виконано*
7.	*Оформлення розділу «Прогнозування аномалій за допомогою алгоритмів машинного навчання*	*14.12-15.12.24*	*Виконано*
8.	*Оформлення розділу «Охорона праці та безпека в надзвичайних ситуаціях»*	*07.12-12.12.24*	*Виконано*
9.	*Нормоконтроль*	*11.12-14.12.24*	
10.	*Попередній захист роботи*	*20.12.24*	*Виконано*
11.	*Захист кваліфікаційної роботи*	*27.12.24*	

Студент _____ *Гладовський Н.О.*
(підпис) (прізвище та ініціали)

Керівник роботи _____ *Баран І.О.*
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Гладовський Н. О. Методи і засоби моніторингу та прогнозування несправностей у розподілених комп’ютерних системах: робота на здобуття кваліфікаційного ступеня магістра: спец. 123 — комп’ютерна інженерія / наук.кер. І.О. Баран. Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2024. — 72 с.

Ключові слова: випадковий ліс, інфокомунікаційна система, моніторинг, прогнозування, усунення несправності, syslog

У кваліфікаційній роботі проведено аналіз сучасних методів та засобів діагностики несправностей комп’ютерних систем. Запропоновано використання сукупності інформаційних систем для проведення діагностики та автоматичного усунення несправностей розподілених комп’ютерних систем. Сформульовано пропозиції щодо застосування поєднання систем технічного обліку, відстеження, оцінювання та системи прогнозування з метою втілення концептуальної моделі автоматизації обслуговування розподілених систем.

Досліджено питання автоматичного усунення несправностей з погляду прогнозування інцидентів з допомогою алгоритмів машинного навчання. Прогноз ґрунтується на вимірах поточного стану комутаторів та історичних випадках відмов апаратного забезпечення комутатора.

Прогнозування несправностей із використанням методу випадкового лісу дозволить завчасно бути готовим до усунення ймовірної несправності, чи, в деяких випадках, навіть і до запобігання цим несправностям. Це дасть змогу ефективно проводити обслуговування розподілених комп’ютерних систем.

ANNOTATION

Hladovskyi N. Methods and means of monitoring and forecasting faults in distributed computer systems. Master’s Graduation Thesis: speciality 123 — Computer engineering / supervisor I.O. Baran. Ternopil: Ternopil Ivan Puluj National Technical University, 2024. — 72 p.

Keywords: random forest, infocommunication system, monitoring, forecasting, troubleshooting, syslog

The Thesis analyzes modern methods and tools for diagnosing faults in computer systems. It is proposed to use a set of information systems for diagnosing and automatically eliminating faults in distributed computer systems. Proposals are formulated for the use of a combination of technical accounting, tracking, evaluation and forecasting systems in order to implement a conceptual model for automating maintenance of distributed systems.

The issue of automatic troubleshooting from the point of view of predicting incidents using machine learning algorithms is studied. The forecast is based on measurements of the current state of switches and historical cases of switch hardware failures.

Forecasting faults using the random forest method will allow you to be prepared in advance to eliminate a probable fault, or, in some cases, even to prevent these faults. This will allow you to effectively maintain distributed computer systems.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ СКОРОЧЕНЬ І ТЕРМІНІВ

SIF (system interconnect fabric) – система міжкомпонентного зв'язку.

БД – база даних.

КС – комп’ютерна система.

ІМ – інформаційна мережа.

ПЗ – програмне забезпечення.

РКС – розподілена комп’ютерна система.

РС – розподілена система.

ВСТУП

Актуальність теми. У зв'язку зі складністю інфраструктури РКС інженери в ході експлуатації технічних систем та обладнання стикаються з безліччю проблем, до яких входять проблеми конфігурації обладнання, проблеми на рівні ПЗ, проблеми та збої апаратного забезпечення. Деякі проблеми важче вирішувати вручну, або ж на їх усунення витрачається багато часу. Крім того, при традиційному обслуговуванні РКС усі дії щодо усунення несправностей найчастіше виконуються вже після того, як виникли негативні наслідки.

Впровадження автоматизованих систем виявлення та прогнозування несправностей та застосування технологій моніторингу дозволить одночасно знизити витрати на обслуговування РКС, а також мінімізувати наслідки збоїв. Тому своєчасний пошук несправностей в КС та проведення раціональних відновлювальних заходів у найкоротші терміни є одним із завдань першорядної важливості з практичної та економічної точок зору експлуатації КС. Дослідження нових підходів до прогнозування несправності дозволить підвищити надійність РКС та оперативність реагування. Для виконання завдання можуть бути використані класифікуючі алгоритми машинного навчання, навчання яких може проводитися на основі історичних даних про несправності за участю експертів.

Мета дослідження: експериментальне дослідження файлів системних журналів, на основі яких проводиться тренування алгоритму машинного навчання з метою прогнозування майбутніх несправностей пристроїв РКС (на прикладі системних журналів комутаторів).

Завдання кваліфікаційної роботи:

– проаналізувати сучасні методи та засоби діагностики несправностей КС;

– сформулювати завдання моніторингу та аналізу рішень систем моніторингу;

– дослідити функціональні та процедурні дії щодо усунення несправностей;

– дослідити взаємодію систем для автоматичного обслуговування РКС;

– застосувати алгоритми машинного навчання для вирішення завдання прогнозування та усунення аномалій в комп’ютерних системах.

Об’єкт дослідження: обслуговування РКС.

Предмет дослідження: автоматизація прогнозування несправностей у РКС.

Методи дослідження: фундаментальні положення комп’ютерної інженерії, метод випадкового лісу, ймовірностні методи з теорії масового обслуговування.

Наукова новизна отриманих результатів:

– запропоновано модель сукупності службових систем для технічного обліку, моніторингу та системи прогнозування. Дана модель дозволяє оптимізувати пошук та прискорити процес відновлення КС у разі їх неполадки;

– отримав подальший розвиток спосіб застосування алгоритму випадкового лісу для пошуку майбутніх несправностей в КС.

Практичне значення одержаних результатів. Полягає у створенні програмних засобів для реалізації моделі прогнозування на основі алгоритму машинного навчання випадкового лісу, що підвищує ефективність усунення несправностей та підвищення якості технічного обслуговування РКС.

Публікації. Окремі результати дослідження апробовано на XII науково-технічній конференції «Інформаційні моделі, системи та технології» як опубліковані тези:

1. Гладовський Н.О. Виявлення несправних параметрів у ході моніторингу мережі // Інформаційні моделі, системи та технології: Праці XII наук.-техн. конф. (Тернопіль, ТНТУ ім. І. Пулюя, 18-19 грудня 2024 р.) с. 119.

2. Гладовський Н.О. Програмні рішення моніторингу систем та мереж // Інформаційні моделі, системи та технології: Праці XII наук.-техн. конф. (Тернопіль, ТНТУ ім. І. Пулюя, 18-19 грудня 2024 р.) с. 118.

Структура роботи. Робота складається з пояснювальної записки та графічної частини. Пояснювальна записка складається з вступу, 4 розділів, висновків, списку використаної літератури та додатків. Обсяг роботи: пояснювальна записка – 74 арк. формату А4, графічна частина – 10 аркушів формату А1.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Аналіз розподілених комп’ютерних систем

Зараз успішно розвиваються різні РС, побудовані на базі зібраного разом об'єднання спеціалізованої обчислювальних засобів і засобів зв'язку, котрі організовують взаємозв'язок інформаційних процесів та пропонують абонентам (користувачам) різноманітну гаму сервісів із обміну інформацією та обробки різних її варіацій. Мережі, котрі забезпечують передавання, опрацювання і зберігання інформації називають ІМ [1].

На витоках розвитку таких ІМ лежала автоматизація окремих елементів інформаційних процесів Були утворені системи збирання, збереження та відшукання інформації із використанням обчислювальної техніки, де центральними процесами власне були зберігання та пошук, але не менш важливими були і процеси опрацювання та передавання даних.

Поруч із розвитком ІМ здійснювалося створення систем передачі та розподілу інформації, у яких головною складовою був процес обміну даними між віддаленими об'єктами. Ці системи називають РС.

До складу РС входять автономні вузли (сервери), що працюють спільно, і виступаючи однією логічною системою. Головною їхньою перевагою є те, що вони полегшують інтеграцію різноманітних застосунків, котрі працюють на різних вузлах, у єдину цілісну систему. Їхня додаткова перевага полягає в тому, що при правильному конструюванні РС чудово масштабуються. Їх розмір визначається лише розмірами базової мережі.

У сфері інфокомунікаційних технологій є різноманітні РС. Розподілені операційні системи застосовуються регулювання апаратним забезпеченням взаємозалежних комп'ютерних систем, куди входять мультипроцесорні і гомогенні мультикомп'ютерні системи. Такі РС насправді складаються з автономних комп'ютерів, але благополучно розцінюються як комплексна система.

Мережеві операційні системи, в той же час, легко пов'язують різні комп'ютери, що працюють під контролем своїх операційних систем, тому користувачі легко можуть отримувати доступ до місцевих служб кожного з вузлів. РКС представлені у вигляді систем, до складу яких входять засоби обробки та зберігання інформації, пов'язані телекомунікаційною мережею, що є єдиною системою. Їм властивий розподіл функцій, ресурсів між робочими вузлами, з чого можна зробити висновок, що вихід з ладу одного вузла не тягне за собою повну зупинку у функціонування системи (рис. 1.1) [2].

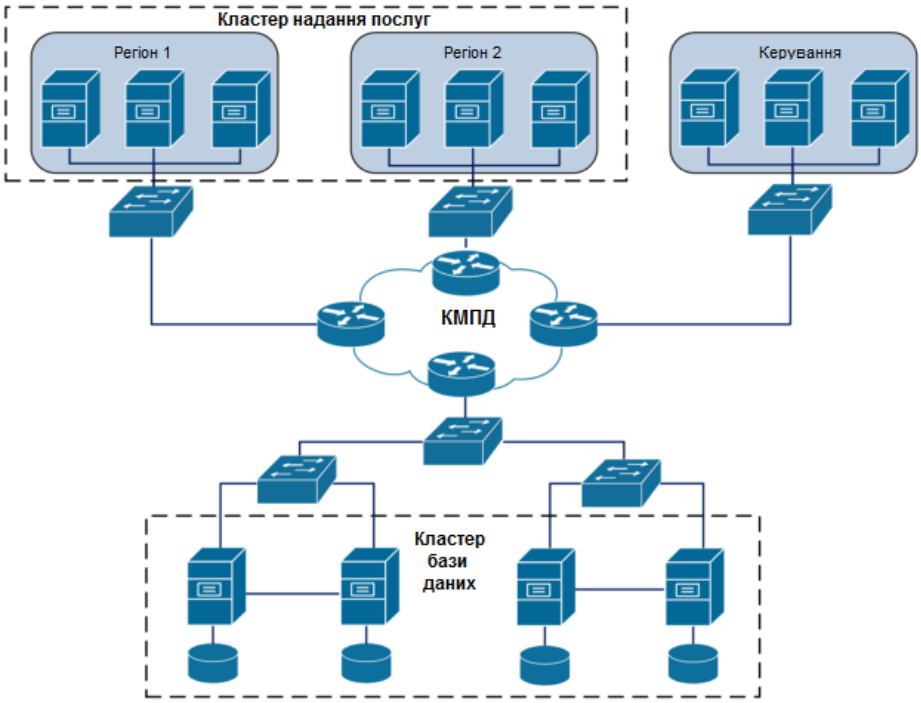

Рис. 1.1. Приклад РКС

Головними критеріями, що висуваються стосовно РС, є: прозорість, відкритість системи, безпека, масштабованість, надійність.

Прозорість, головним чином, полягає в тому, що РС повинні прийматися користувачами як однорідний об'єкт, а не комплекс самостійних об'єктів, які взаємопов'язані один з одним. Проектування РС є складним завданням і дуже важливо дотримуватись необхідної прозорості, що є важливою умовою функціонування системи. Існують різні види прозорості.

Прозорість розташування у РС у тому, що користувачеві не потрібно

знати, де містяться необхідні йому ресурси. Файли можуть пересилатися на різні вузли РС. Якщо на вузлі РС виник збій, і дані були відновлені на іншому вузлі РС, разом з тим користувач не повинен вловити ці переміщення. Наприклад, у розподілених інформаційних файлових системах, користувачеві необхідно бачити лише єдиний файловий простір, при цьому дані можуть розташовуватися на різних серверах.

Принцип прозорості доступу один із основних принципів у РС. Прозорість, таким чином, полягає у наданні приховування відмінностей доступу та забезпеченні даними. У зв'язку з цим необхідно забезпечити паралельне використання ресурсами системи, тим самим забезпечити приховування факту спільного користування ресурсами.

Для забезпечення збереження даних, зокрема на розподілених файлових системах, важливо забезпечити резервне копіювання даних. Користувачеві не потрібно знати, що існує резервне копіювання даних. Для приховання даного показника, необхідно, щоб дані або ресурси, що надаються, мали однакові імена.

У порівнянні з ранніми РС, які по суті були обмеженими та закритими, оскільки вони були створені в основному в рамках окремих організацій та з метою вирішення конкретних завдань, сучасні РС створюються найактивнішими. Використання принципу відкритості до РС стало доступним через розвиток ліній передачі даних, збільшення продуктивності процесів, а також прогрес у телекомунікаціях та інформаційних технологіях. Відкритість РС виявляється у можливості взаємодії з іншими відкритими системами. Відкритим системам притаманні такі характеристики:

– ці системи мають відповідати чітко позначеним інтерфейсам;

– системи, що належать до РС, мають безперешкодно взаємодіяти між собою;

– системам необхідно надавати переносимість застосунків.

Відкритість системи забезпечується за допомогою: мов програмування, апаратних платформ, ПЗ.

Безпека систем займає одне з найважливіших місць у сучасних РС. Безпека

РС є, загалом, комплексом трьох критеріїв.

Потреба у створенні РС, що забезпечують необхідну безпеку даних та всього складу РС, з'являється у всіх сферах. Більшість питань безпеки вирішуються лише на рівні окремих вузлів РС, наприклад, з допомогою встановлення фаєрволів і антивірусного ПЗ на різні вузли системи, з допомогою політики розпізнавання користувачів та інші методами. Але у зв'язку зі специфікою архітектури більшості РС цей метод не завжди вважається ефективним. ПЗ не завжди може дотримуватися необхідної конфіденційності даних у РС. Наприклад, ПЗ будь-якого обладнання який завжди може забезпечити повноцінний захист від DDoS атак на розподілену мережу. Нерідко способи захисту від схожих атак не завжди підходять для вузлів обчислювальної мережі. Ступінь доступності РС проявляється не тільки в доступності до ресурсів у момент часу , а й методами організації захисту РС, оскільки, переважно, програмні засоби, які забезпечують захист від атак, спрямовані на відмову в обслуговуванні. Особлива увага до цього питання приділяється багатьма розробниками антивірусного ПЗ.

Внаслідок появи нових методів і алгоритмів, чутливих до обчислювальних ресурсів і, найважливіше, до ресурсів часу, потреба у доступності РС у час t стає максимально актуальною. Однією з найважливіших показників надійності всієї РС є відмовостійкість, головний показник обчислювальної системи, який ґрунтується на можливості продовження дій, заданих програмою, після виникнення збоїв.

Всупереч усім перевагам РС у порівнянні з традиційними централізованими системами, а саме РС пропонують на порядок меншу вартість розгортання та простоту реалізації, у таких системах є чимало вагомих недоліків. Головними проблемами РС у порівнянні із традиційними системами можна назвати:

– недоліки в адмініструванні системи;
– проблеми обмеженості масштабованості РС;
– проблеми переносимості ПЗ.

До проблем адміністрування системи можна віднести:

– проблеми балансування навантаження на вузли системи;

– проблеми відновлення даних у разі виникнення несправностей та аномалій.

Поділ ресурсів у РС передбачає необхідність створення гнучких засобів адміністрування, що налаштовуються. Оскільки в глобальних РС адміністрування має відбуватися в автоматичному режимі, тому з'являються такі основні проблеми адміністрування РС:

– балансування навантаження на вузли системи;

– відновлення даних у процесі виникнення помилки;

– збір статистичних даних із вузлів системи;

– оновлення ПЗ на вузлах системи в автоматичному режимі.

Необхідно врахувати, що вказаний список є узагальненим, тобто для кожної окремої РС має місце виникнення непередбачених проблем. Втім, ці проблеми найчастіше зустрічаються практично і варті особливої уваги розробки РКС. Дві останні проблеми є найбільш вивченими, і на ринку ПЗ РС є множина розроблених програмних засобів, що забезпечують як збір статистичних даних, так і оновлення ПЗ. Зі своєї сторони методи балансування навантаження на вузли системи та методи відновлення даних у разі виникнення помилок становлять науковий інтерес. Завдяки своїй специфіці, а також широкому спектру обладнання та архітектури РКС у сфері телекомунікації немає єдиного методу проектування, для кожної конкретної РС можливе виникнення непередбачених проблем. Однак дані проблеми найбільш часто зустрічаються на практиці і заслуговують на особливу увагу при проектуванні РКС.

У зв'язку зі складністю КС розробники стикаються з безліччю проблем, і деякі з них складніше вирішувати «вручну», або ж на усунення несправностей йде багато часу. Крім цього, під час традиційного обслуговування РКС усі маніпуляції щодо усунення несправностей або «вузьких місць» робляться вже після того, як виникли негативні наслідки.

Раціональна технічна діагностика є одним із найбільш ефективних методів покращення надійності об'єктів телекомунікації у процесі експлуатації,

що дозволяє організувати пошук та виявлення несправності. Важливого значення ця проблема набуває при використанні та обслуговуванні складних систем спеціального призначення, для яких основними показниками якості процесу діагностики є надійність встановлення діагнозу та час пошуку дефектних компонентів. Крім цього, раціональним рішенням цього завдання є розробка та впровадження систем моніторингу і діагностики техпроцесу і обладнання, фукціонування яких ґрунтується на використанні новітнього ПЗ, високоточного вимірювального обладнання, а також експертних систем, що здійснюють підтримку прогнозування за допомогою методів машинного навчання при діагностуванні РКС.

1.2. Аналіз програмних рішень систем моніторингу

Системи моніторингу дозволяють усувати незначні несправності, що становлять основну масу інцидентів, автоматично або за допомогою підключення віддаленим доступом до обладнання, що адмініструється. При усуненні значних несправностей системи моніторингу якраз і призначені для їх недопущення або, в крайньому випадку, оперативного усунення. Для таких цілей проводиться аналіз та локалізація причин несправностей на основі даних зібраної статистики та даних прогнозування. При побудові архітектури високої доступності, яка передбачає наявність в інфраструктурі РКС дублюючого або резервного обладнання, система може перерозподіляти навантаження на резервне обладнання, забезпечуючи безперебійність надання сервісу або послуги у разі несправності кінцевих користувачів. Розглянемо найпопулярніші системи моніторингу.

Система моніторингу Cacti має високу швидкість розгортання при малій кількості об'єктів для моніторингу. Додавання до моніторингу пристроїв до стандартних шаблонів відбувається швидко і легко. Але процес додавання пристроїв з метриками, які не передбачені стандартним набором функцій, трудомісткий. Як сховища даних Cacti використовує RRD (Round-Robin Database), а як візуалізатор для побудови графіків із зібраних даних - утиліту RRDTool [9].

Це недолік, оскільки RRD усереднює дані, і неможливо сказати, яке було точне значення параметрів кілька місяців тому. Відсутня система оповіщень, основні функції - збір метрик із пристроїв у RRD та їх подальша візуалізація. На рис. 1.2 представлений приклад графіка утилізації пропускної спроможності мережного інтерфейсу комутатора, побудований за допомогою Cacti.

Рис. 1.2. Графік утилізації пропускної спроможності, побудований з допомогою Cacti

Головний показник продуктивності моніторингу - кількість значень за секунду, оброблених без затримок у реальному часі

$$VPS = \sum_{i=1}^{n} \sum_{j=1}^{m_i} \frac{l}{t_i},$$

де VPS (values per second) – сумарне число значень за секунду; n - загальна кількість пристроїв моніторингу; m_i - кількість метрик, що збираються із сервера; t_{ij} – періодичність збору метрики *j* сервера *i*, с

У таблиці 1.1 наведено характеристики розглянутих систем моніторингу.

Таблиця 1.1

Порівняльні характеристики деяких систем моніторингу

Характеристика	New Relic	Graphite	Nagios	Zabbix
Кількість серверів /метрик без затримок	10^4 / 10^5	10^3 / 10^5	400 / 10^3	10^5 / 10^6
Оплата ліцензії	Платно	Безкоштовно	Безкоштовно	Безкоштовно
Налаштування конфігурації моніторингу	Легко: вбудовані шаблони	Важко: кодування надбудови	Середньо: необхідна адаптація	Легко: вбудовані шаблони
Мови програмування надбудов	Ruby, Java, PHP, Python, .Net	Python, C, PHP, Ruby, .Net, Java	C, Perl, Python, Shell scripts	Будь-який
Типи СУБД, що підтримуються для зберігання даних моніторингу	Sharded MySQL, Perconal build	Whisper	MySQL	MySQL, PostgreSQL, Oracle, IBM DB2, NoSQL
Контрольовані ОС віддалених серверів	Windows, Linux	Windows, Linux, MacOS	Windows, Linux, FreeBSD	Windows, Linux, MacOS, FreeBSD, IBM AIX
Прогнозування	Так	Так	Так	Так
Автоматичне відновлення	Так	Ні	Так	Так

1.2.1. Graphite. ПЗ застосовує тільки свою БД Whisper, а також є складним для налаштування, це є вадою [10]. Проте є компенсація за рахунок високої продуктивності за великих даних обробки - 10 VPS з 10 серверів у реалі. Компаніся Orbitz створила Graphite для сосбистого моніторингу системних та бізнес-метрик, котрі реалізовані із використанням надбудов, у т.ч. і для прогнозування. Інтерфейс рішення Graphite відображено на рис.1.3.

Недолік продукту полягає у специфіці реалізації функцій під чітко

визначені бізнес-процеси. Ще одним недоліком є відсутність автоматичного відтворення сервісів вслід за виявленням відмови. В цілому існують суб'єктивні та суперечливі думки про незручний інтерфейс додатків Graphite та вимушено високі витрати дуже великої кількості моніторингових серверів Graphite.

Рис. 1.3. Інтерфейс системи моніторингу Graphite

1.2.2. New Relic. Більше відома як програмний засіб відстеження та оцінювання застосунків, проте містить рішення також і для системних індексів із вбудованим сприянням фільтрації помилкових повідомлень та передбачення відмов (рис. 1.4). За даними [11], біля 20 млн осіб застосовують розробки New Relic, котрі є доступними у хмарних середовищах, зокрема Amazone і Azure, завдяки чому стали популярними.

Проте, безпекові умови компаній не дають змогу збереження даних, хоча б у зашифрованих, на зовнішніх хмарних сховищах. Для вирішення цієї проблеми New Relic дозволив під’єднювати свої модулі через JSON HTTP POST, і одержувати дані від серверів New Relic за допомогою API функцій та будувати власні екрани візуалізації даних. У роботі [12] містяться більш детальні відомості про архітектурні особливості New Relic та практичні рекомендації щодо моніторингу SaaS додатків з великими потоками даних таких як сучасна мережа X.

Рис. 1.4. Інтерфейс New Relic

1.2.3. Nagios. Система моніторингу Nagios, яка має архітектуру з відкритим кодом, призначена для моніторингу КС, комп'ютерних систем та мереж (рис. 1.5).

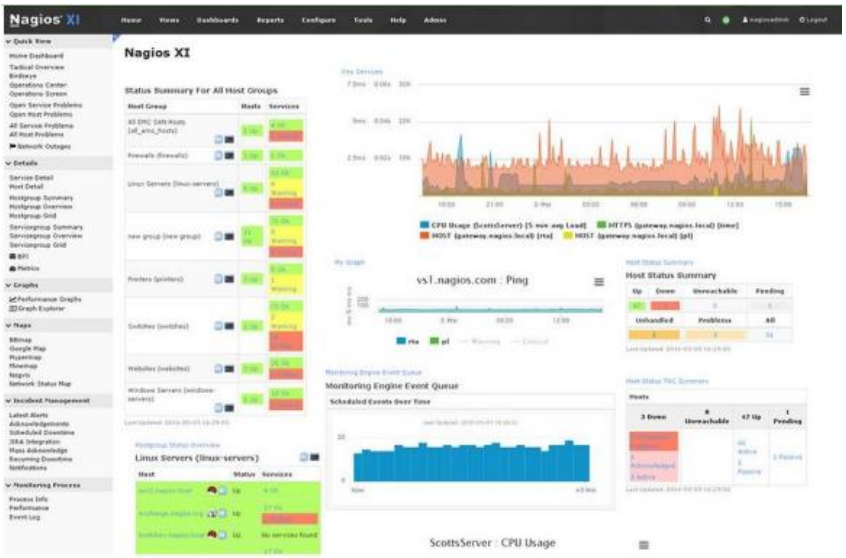

Рис. 1.5. Інтерфейс системи моніторингу Nagios

Nagios проводить моніторинг за введеними вузлами та сервісами, та повідомляє мережевого адміністратора у разі, якщо деякий сервіс чи вузол припиняють (або відновлюють) свою роботу. Система моніторингу передбачає можливість створення карти мережі, із зазначенням детальної інформації про аналізовані об'єкти.

Особливості продукту:

– моніторинг стану робочих станцій (завантаження процесора, застосування простору жорсткого диска);

– інтуїтивно зрозуміла архітектура модулів розширень дає змогу застосовуючи будь-яку мову програмування, досить просто і легко створювати свої способи перевірки сервісів;

– є можливість інтеграції з іншими системами моніторингу для покращення надійності і побудови розподіленої системи відстеження та оцінювання.

До недоліків Nagios можна віднести:

– відсутність системи, що дозволяє знімати дані та довго їх зберігати;

– відсутність можливості самостійно аналізувати отримані дані;

– складність налаштування Nagios: відсутність готових шаблонів саме під конкретні завдання чи системи. Слід бути готовим до того, що, швидше за все, при початковому запуску системи можлива поява множини помилок та попереджень.

Планування та розробка систем моніторингу є досить дорогими заходами, з метою виконання яких можна застосовувати готові продукти та рішення, які будуть справлятися з необхідними завданнями. Однак налаштування та обкатка цього комплексу неминуче призведуть до зайвих витрат часу та коштів, що зрештою й визначить вартість запуску системи. Система Nagios припускає наявність чинника фактично абсолютної переносимості системи. Іншими словами, один раз налаштувавши цю систему або розібравшись з її налаштуванням за допомогою компаній-інтеграторів, можна не турбуватися про те, що зусилля зникнуть при черговому оновленні обладнання сервера. Тим не менш, Nagios не має стійкої до відмови конфігурації і для проведення масштабування, необхідно проводити перенесення конфігураційних файлів, деякі з яких мають великий обсяг, що є недоліками системи.

1.2.4. Prometheus. Є ще однією системою моніторингу різних сервісів та систем, при допомозі котрої сисадміни здатні збирати дані щодо метрик систем,

налаштовувати додаткові функції для налаштування додаткових повідомлень (наприклад, повідомлення про те, коли виникають проблеми). Prometheus здобув популярність серед розробників, оскільки це проект з відкритим вихідним кодом, більшість компонентів якого написана мовою програмування Golang, а частина мовою Ruby. Виходячи з цього можна вивести, що у користувача даною системою буде лише один бінарний файл, який необхідно буде завантажити і встановити разом з компонентами системи Prometheus, в кінцевому результаті розгортання система виглядатиме за прикладом на рис. 1.6.

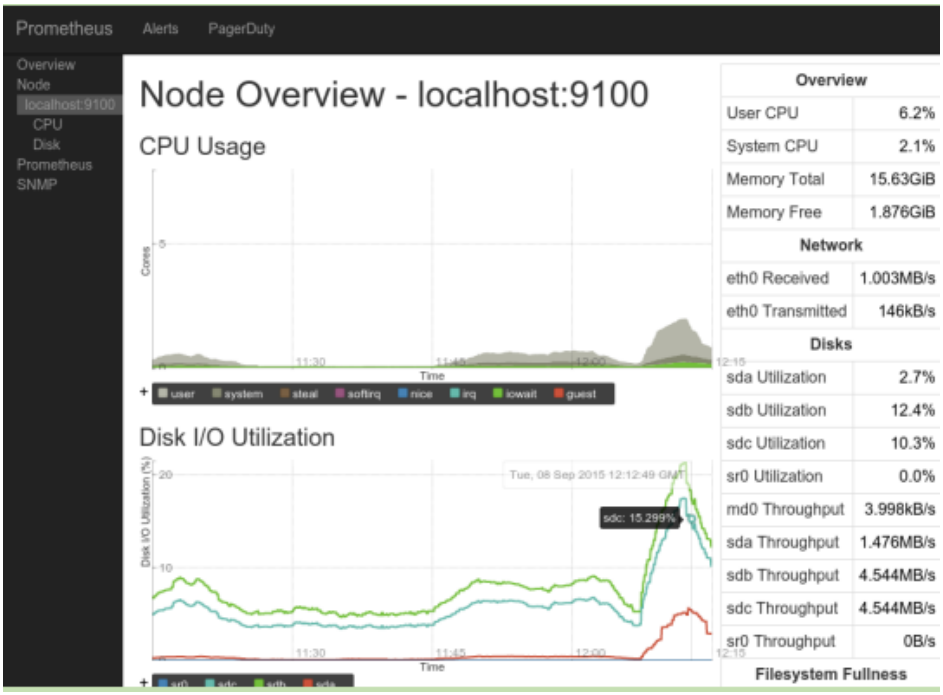

Рис. 1.6. Головне вікно системи моніторингу Prometheus

Вирішальним компонентом системи моніторингу Prometheus є prometheus server, який працює автономно, не залежить від підключення до мережі, і зберігає дані моніторингу на локальній машині. Основне завдання компонента є зберігання та моніторинг певних об'єктів. Сервіси для моніторингу визначаються та завантажуються автоматично, за допомогою пошукового механізму та установок за замовчуванням, зроблених розробниками рішення. Розгортання системи моніторингу відбувається досить швидко за цим принципом. У результаті для того, щоб налагодити процес моніторингу певної системи, достатньо встановити

сервер, щоб ввести процес і систему моніторингу в експлуатацію [13].

Дані в Prometheus представлені у вигляді тимчасових рядів. Мітки часу мають точність до мілісекунд, значення представлені з 64-бітною точністю. Варіативна мова запитів дає можливість знайти і при необхідності зберегти вибірково будь-яку інформацію з отриманої вибірки. На основі набору даних можна будувати графіки чи генерувати проблемні білети. Крім цього, при підключенні бібліотек користувача сервер може отримувати інформацію від систем, які підключаються в мережу періодично.

Prometheus є децентралізованою самоврядною архітектурою, при такому підході є можливість проводити моніторинг сотень серверів з одного місця. Причому за потреби на деяких системах може бути незалежний сервер моніторингу Prometheus. Дана система є структурованим комплектом інструментів, що складається з метрик для агрегації, візуалізації та генерації повідомлень.

1.2.5. Zabbix. Універсальна система моніторингу, що дозволяє спостерігати різні параметри на віддалених машинах. Параметри заносяться в БД, які при необхідності можна витягти, наприклад, для побудови графіків, налаштування оповіщення при переході значень параметрів через межу та ін. який знаходиться під моніторингом, опис проблеми, тривалість несправного стану та резолюція щодо усунення несправності (рис. 1.7). Оповіщення можуть здійснюватися як електронною поштою, так і через SMS.

Рис. 1.7. Журнал повідомлень про несправності у системі моніторингу Zabbix

До основних можливостей програмного продукту можна віднести [14]:

– з робочих станцій збираються та аналізуються на стороні сервера кількісні характеристики роботи мережі, а не результати перевірок (на кшталт «зламалося чи ні»);

– сценарії на основі моніторингу, комплексна реакція на події;

– підтримка високопродуктивних агентів практично для усіх платформ;

– розширення функціональності з допомогою виконання зовнішніх програм;

– можливість створювати логічні схеми мереж

– єдина точка збирання інформації для користувачів;

– час зберігання даних обмежений лише дисковим простором.

Zabbix є найбільш гнучким рішенням, яке адміністратор отримує комплексно, без встановлення додаткового стороннього ПЗ. Суть та переваги системи, перераховані вище, можливі завдяки трьом важливим властивостям: Item (Айтем), Trigger (Тригер) та Template (Шаблони). Перший параметр, який відстежується на робочій станції. Айтемом позначається обсяг вільного місця на диску, температуру процесора, контрольну суму файлу або якесь довільне число, що повертається функцією користувача. Також в айтемі задається інтервал його опитування за секунди. Це дозволяє більш тонко відбудувати систему моніторингу: найважливіші параметри опитувати частіше, менш важливі – рідше. Тим самим знижується навантаження як на сервер (який може опитувати сотні машин та тисячі параметрів), так і на робочі станції, які можуть бути малопродуктивними. Тригер - поріг айтема, перехід через який спрацьовує системи моніторингу. У тригері можна вказати як значення та умову (більше, менше), так і час (миттєве значення, максимальне, мінімальне або середнє за період у секундах). Для тригера можна задати пріоритет і на його основі запрограмувати дії системи моніторингу на кшталт: «не застосовувати будь-які дії», «надіслати повідомлення по e -mail», «надіслати SMS», «виконати операцію» і т.д. У web -інтерфейсі є кілька різних способів наочно подивитися поточний стан тригерів: як усіх, так і тих, що тільки спрацювали. У Zabbix тригери мають 6 рівнів пріоритету, що дозволяє адекватно доносити оповіщення про різні події. За

допомогою шаблонів можна створити типову конфігурацію, додати до нього айтемів і тригерів і підключити цей шаблон до робочої станції. Таким чином, можна створювати з типових шаблонів працездатні конфігурації моніторингу. За замовчуванням у системі є кілька стандартних шаблонів для найпопулярніших платформ, додатків, протоколів та ін.

1.3. Висновки до розділу

У першому розділі здійснено аналіз сучасного стану питання обслуговування КС.

Окрім цього було проведено аналіз існуючих рішень системи моніторингу, визначено переваги та недоліки кожної системи.

РОЗДІЛ 2
ТЕОРЕТИЧНА ЧАСТИНА

2.1. Завдання моніторингу

У ході експлуатації КС поряд з адмініструванням, розподілом ресурсів, управлінням взаємодією різних систем або підсистем між собою інженери стикаються з безліччю проблем, і деякі проблеми важче вирішувати «вручну», або ж на усунення несправностей витрачає багато часу. РКС мають архітектуру підвищеної складності завдяки територіальній розподіленості інфраструктури, наявності підсистем захисту інформації та підтримки інформаційної безпеки, а також наявності резервування даних та дублюючих елементів, які забезпечують безперебійну роботу та надійність систем. За таких параметрів специфічним завданням є підтримка параметрів роботи КС на заданому рівні. Першою та основною умовою роботи систем є підтримка мережної інфраструктури в робочому стані. Це вимагає численного штату висококваліфікованих фахівців, обов'язки яких полягають в оптимізація роботи мережі, підтримці доступності сервісів, менеджмент і своєчасне оновлення мережного апаратного та ПЗ, усунення та виявлення аномалій та інцидентів, що з'являються, грамотна технічна підтримка в рамках своєї зони діяльності та компетенції, та інші аналогічні функції. Не виключаються і складнощі у процесі роботи з КС. По-перше, фахівці, зазначені раніше, повинні чітко мати уявлення про цілу низку технологій, а також мати навички роботи з ними, що підвищує планку до необхідних критеріїв фахівців та рівня підготовки працівників. По-друге, у зв'язку з невисоким рівнем оплати праці залучення подібних фахівців працювати в організаціях стає трудомістким завданням. По-третє, існують бюджетні та нормативні обмеження, згідно з якими існує поріг для загального числа найманого штату співробітників і необхідно дотримуватись цього порога.

Для більшої частини описаних проблем і для загального розуміння того, як дізнатися про проблему використовуються системи моніторингу. Такі системи

служать для:

– безперервного моніторингу стану та функціонування використовуваного обладнання, прикладних систем та сервісів;

– збору статистичних даних (в реальному часі та у вигляді архівів) та забезпечення наочного подання важливих метрик продуктивності обладнання;

– визначення інцидентів;

– проведення аналізу впливу виявлених несправностей;

– автоматичного повідомлення про виявлені інциденти, відповідальних осіб за їх усунення або локалізації причин інциденту.

Розглянемо постановку завдання моніторингу комп'ютерної мережі. Загалом завдання моніторингу можна описати в такий спосіб. Події, що відбуваються в комп'ютерних мережах та інформаційних системах, мають випадковий характер. У зв'язку з цим їх вивчення найбільш підходящими є ймовірнісні математичні моделі теорії масового обслуговування, наприклад теорія марковських ланцюгів [8].

Для моделювання спочатку необхідно визначити вхідні дані. Розглянемо події, пов'язані з детектуванням аномалій у процесі діагностики та контролю досліджуваних системою параметрів обладнання в мережі, як стан аналізованої системи моніторингу мережі.

Виходячи з вищевикладеного, виведемо можливі стани в процесі моніторингу метрик (параметрів) мережного обладнання: S_0 – відсутність несправностей; S_1 - навантаження мережі; S_2 – зниження пропускної спроможності; S_3 - відсутність мережного порту; S_4 – фізична недоступність пристрою; S_5 - фрагментація пакета; S_6 - повна відмова системи.

Описаний ланцюг подій під час моніторингу мережі можна подати у вигляді графа станів, як показано на рис. 2.1.

Імовірність виявлення несправного параметра в мережі визначається в даному випадку як ймовірність знаходження системи в стані S_i.

На графі кожна стрілка відповідає інтенсивності потоку подій, який переводить систему з одного стану в інший. Послідовність опитування

визначається призначеним системним адміністратором пріоритетом [17]. Чим вищий пріоритет у параметра мережного пристрою, тим вище інтенсивність його опитування. За інтенсивності переходів у разі приймається I_i - інтенсивність опитування i -го параметра у системі моніторингу.

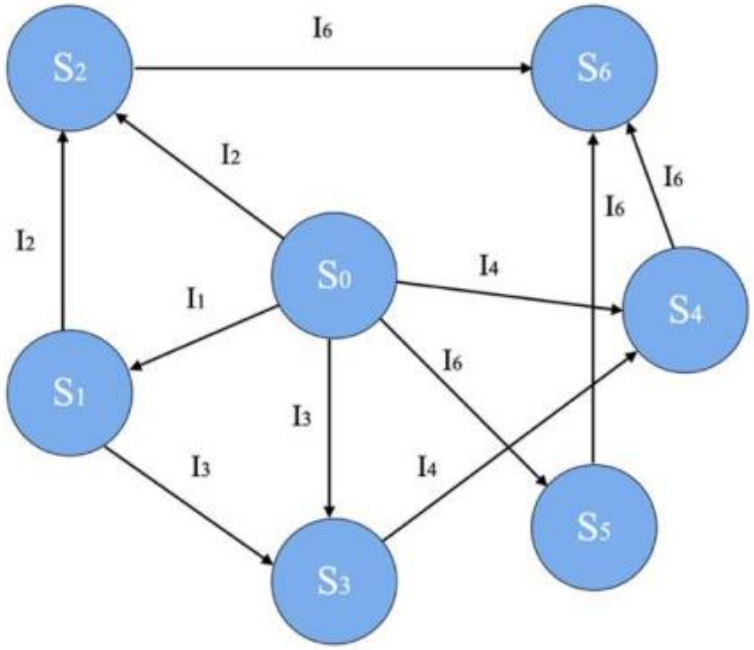

Рис. 2.1. Граф виявлення несправних параметрів у ході моніторингу мережі

Зважаючи на описані вище метрики роботи, система моніторингу виходить, що вона зі стану S_0 при виконанні опитування мережевих пристроїв при відмові будь-якого компонента переходить у стан виявлення несправностей S_i з інтенсивністю I_i. За графом можна скласти математичну модель процесу моніторингу мережі з урахуванням ймовірностей p_i як системи рівнянь. Її розв'язок дозволяє відстежити динаміку діагностування несправних параметрів у процесі мережного моніторингу шляхом відстеження ймовірностей у певні проміжки часу.

2.2. Принциви усунення несправностей під час обслуговування систем

Відповідно до [19], усунення несправностей включає три етапи:

– інспекція (оцінка ефективності системи шляхом оцінки змін у

характеристиках виходів чи компонентів системи);

– пошук та усунення несправностей, пошук компонентів системи, що викликають негативний висновок;

– пошук дій, які виправлять невідповідність (причинно – поведінкові зв'язки чи виправлення).

Однак дана концепція не торкається ролі попереднього досвіду, який є найчастішою стратегією діагностики несправностей. Досвідчені фахівці або експерти з усунення несправностей найбільш ефективні, тому що вони мають уявлення про необхідні дії, що ґрунтуються на проблемах, які вони вирішували раніше.

Побудова проблемного простору є першим кроком у вирішенні проблем [20] під час обслуговування будь-яких систем. Проблемним простором будь-якої проблеми при усуненні неполадок є знання та представлення моделі представленої задачі перед інженером.

Після створення проблемного простору інженер розпочинає процес діагностики, досліджуючи несправну систему та порівнюючи стани системи з аналогічними проблемами, які він вирішував раніше. Якщо згадати попередню проблему, простір проблеми відразу зменшується, щоб увімкнути опис старої проблеми.

Упродовж усього циклу «генерації та перевірки гіпотез» фахівці з усунення несправностей намагаються ще більше звузити проблемний простір та ізолювати потенційні несправності [21]. Виділяють чотири типи гіпотез:

– системні гіпотези припускають помилку на системному рівні;

– підсистемні гіпотези припускають помилку лише на рівні підсистеми і зменшують проблемний простір до конкретної підсистеми у межах системи;

– гіпотези пристроїв припускають помилку на рівні пристрою та зменшують проблемний простір до обмеженої кількості компонентів у підсистемі;

– гіпотези компонентів - найбільш конкретний тип гіпотез, які припускають помилку на рівні компонента та призводять до ідентифікації окремого компонента як потенційна причина помилки.

Коли всі потенційні гіпотези створені, ці гіпотези мають бути перевірені та оцінені. Процес виділення несправності є пошук по всій системі від підсистем, пристроїв до компонентів ієрархічним чином, щоб визначити причину несправності.

Інженер повинен вибрати найбільш правильне рішення з набору згенерованих рішень та визначити, яке з них найкраще відповідає всім обмеженням (наприклад, ефективність, результативність, специфіка системи або економічна міркування). Недосвідчені фахівці з усунення неполадок часто впроваджують, а потім перевіряють ефективність різних рішень. На підставі результатів тесту недосвідчений фахівець із усунення несправностей приймає або відхиляє обране рішення. Це не найефективніший спосіб усунення несправностей. Досвід роботи має усунути необхідність повторного тестування.

2.3. Методи вирішення задачі автоматизації обслуговування

Практично всі бізнес-процеси великих організацій підтримуються інфокомунікаційними технологіями, тому незначні порушення у роботі РКС можуть викликати простої в роботі організації або погіршити якість послуг, що надаються клієнтам, що, як правило, спричиняє суттєві фінансові втрати. Тому своєчасна діагностика та локалізація несправностей, а також максимально швидке відновлення працездатності компонентів РКС є одним із найважливіших завдань у сфері телекомунікації.

Для виявлення несправностей визначається стан та якість функціонування елементів РКС, після чого значення параметрів роботи елементів, підсистем тощо порівнюються з пороговими значеннями.

У [22] розглядаються проблеми функціонального моніторингу з використанням порогових схем, однак при прийнятті рішень беруть участь лише прості порогові схеми без гістерезису. Схеми прийняття рішень, запропоновані [23] і апробовані в [24], використовують локальні механізми гістерези, що дозволяють істотно скоротити кількість спрацьовувань вирішальних схем, але при

цьому не розглядаються питання, пов'язані з визначенням значень порогів, що встановлюються.

Для виявлення несправностей у комп'ютерних мережах пропонується використовувати статистичний, або реактивний моніторинги. При цьому не розглядаються питання використання цих видів моніторингу у взаємозв'язку.

У телекомунікаційних системах використовуються два основних методи - опитування та передача повідомлень про події [25], а інші методи, як правило, не розглядаються. У той же час використання цих методів не дозволяє забезпечити ефективний моніторинг IT-систем.

Для пошуку та локалізації несправностей у телекомунікаційних системах зазвичай використовують два підходи: пасивна діагностика та активні перевірки. У першому випадку відбувається пасивний збір інформації про стан системи, яка потім обробляється для знаходження джерел несправностей. При пасивному підході втручання у роботу системи мінімальне, проте, пошук першоджерела несправностей може зайняти тривалий час за наявності великого відсотка втрачених симптомів. Активний підхід передбачає використання тестових перевірок визначення несправностей. До недоліків цього методу слід віднести значне втручання у роботу системи, і навіть неможливість виявлення деяких несправностей з допомогою перевірок.

У [26] розглядається пасивний метод локалізації проблем у мережах з використанням матриці ймовірностей, що пов'язує симптоми з можливими несправностями. Метод дозволяє ранжувати набір потенційних несправностей, відрізняється низькою ймовірністю отримання помилкових симптомів, проте не виключає помилкові висновки і вимагає великого часу аналізу для великомасштабних мереж.

У [27] описаний механізм кореляції для багаторівневої діагностики несправностей, заснований на використанні ієрархічної моделі і що дозволяє здійснювати кореляцію повідомлень про несправності в мережі та роботі розподілених додатків. При цьому не враховується можлива втрата симптомів, що суттєво знижує точність результатів.

Запропонований у [28] метод пошуку несправностей у мережах за допомогою активних перевірок, що вибираються з урахуванням результатів попередніх перевірок, відчуває труднощі з виявленням збоїв та аномалій при зміні параметрів функціонування.

Метод генерування набору активних транзакцій виявлення джерела порушення функціонування в РКС, запропонований в [29], зосереджується на пошуку рідкісних видів несправностей.

У [30] запропоновано метод локалізації несправностей в інтернет-мережах та об'єднаних мережах, з використанням одержуваних пасивних симптомів та активних перевірок. На підставі пасивно зібраних симптомів за допомогою матриці, що відображає взаємозв'язки між симптомами та несправностями, виводяться гіпотези про наявність у мережі певних несправностей. Проводиться оцінка вибраних гіпотез і, якщо вони пояснюють всі симптоми і не породжують симптомів, які не спостерігаються в мережі, видається повідомлення про джерела несправностей. В іншому випадку здійснюється пошук нових симптомів або виявлення недостатніх для підтвердження гіпотез за допомогою активних перевірок. Однак у складній системі з великою кількістю елементів збір та обробка інформації про взаємозв'язки між несправностями та їх симптомами є складним та ресурсомістким завданням. Крім того, у масштабах усієї системи безліч різних несправностей можуть викликати однакові або схожі симптоми, що робить результати локалізації несправностей менш точними і збільшує кількість хибнопозитивних результатів. Тому використання матриці несправність-симптом для локалізації несправностей є раціональним лише для окремих класів об'єктів, які можуть відповідати як окремим елементам розподіленої системи, так і її підсистемам, наприклад, підсистемі документообігу, електронної пошти тощо. Класифікація об'єктів проводиться залежно від функціональності об'єкта, типізації його елементів та взаємозв'язків усередині функціональної чи технологічної підсистеми. При цьому враховується ступінь впливу елементів та зв'язків на якість функціонування підсистеми загалом. Доцільно виділення наступних класів.

Клас «Сервер – мережа – користувачі». Такому класу відповідає, наприклад, технологічна підсистема – електронна пошта. У цьому випадку на сервері працює програма, яка надає послуги користувачам. З погляду функціонування підсистеми, зв'язок з окремим користувачем не є критичним, тобто відсутність цього зв'язку ще не є несправністю (користувач може самостійно відключитися від сервера тощо). З метою усунення несправностей перш за все необхідно стежити за працездатністю сервера та його доступністю через мережу. Для даного класу симптомами несправностей будуть недоступність сервера або погіршення параметрів його функціонування.

Клас «Сервер – мережа – додаток». Цьому класу відповідає, наприклад, функціональна підсистема. При цьому на сервері працює програма, яка виконує обчислення та/або аналіз на основі даних та рішень, що надходять із програми. За відсутності зв'язку з програмою надсилається повідомлення про несправність. Також важливою є робота сервера та окремих додатків. Для даного класу симптомами несправностей будуть відсутність одного з елементів підсистеми, погіршення параметрів їх роботи.

Клас «Сервер – мережа – сервер». Цьому класу відповідає, наприклад, взаємодія сервера додатків із сервером БД, коли на сервері працює програма, яка надсилає запити БД. Зв'язок між сервером і БД є критичним для функціонування підсистеми, так само, як і працездатність сервера та БД.

Можна ввести класи, які містять резервні сервери або інші елементи. У цьому випадку при недоступності або несправності основного елемента здійснюється автоматичне перемикання на резервний. Повідомлення про несправність основного елемента є першочерговими, як і враховується на етапі класифікації повідомлень про несправності за рівнем важливості.

2.5. Взаємодія систем для автоматичного обслуговування РКС

Структурна схема взаємодії систем для автоматичного обслуговування РКС [31] представлена на рис. 2.2.

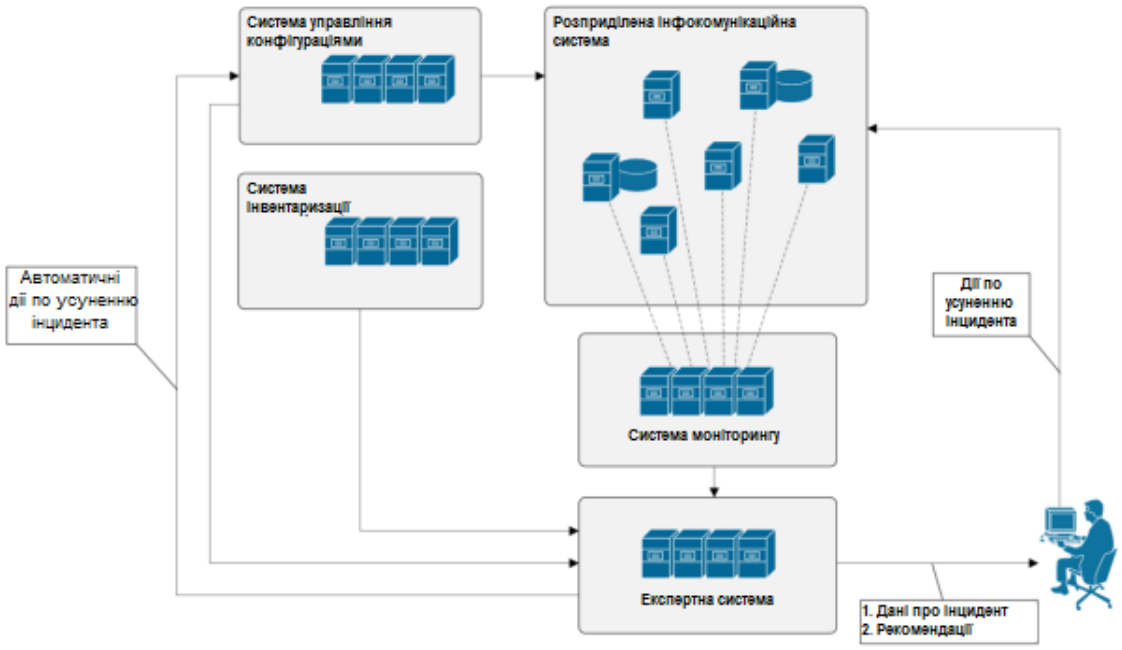

Рис. 2.2. Схема сукупності систем для автоматизації усунення несправностей

Для забезпечення працездатності РКС застосовуються різні автоматизовані системи моніторингу, описані в п.1.2, що полягають у постійному спостереженні та періодичному аналізі об'єктів системи, з відстеженням динаміки змін, що відбуваються з ними.

Варто зважати на той факт, що системи моніторингу не дозволяють зрозуміти, чим викликана проблема і як з нею боротися. Вони можуть здійснювати збирання даних, візуалізувати ці дані. Однак система моніторингу є одним із елементів у процесі автоматизації обслуговування.

Наступним кроком у процесі автоматизації обслуговування буде кореляція отриманих діагностичних даних. Для цього завдання знадобляться система інвентаризації та система управління конфігураціями.

Оскільки система моніторингу має обмежену інформацію про ІК, що спостерігається, цих даних може бути недостатньо. Системи інвентаризації містять інформацію про різні елементи КС та інших систем, що знаходяться у виробничій експлуатації на підприємствах. Для повної картини аналізу інциденту може знадобитися інформація про ідентифікатор елемента КС, його розташування, про його підключення до інших елементів, термін дії ліцензій

тощо.

Система керування конфігураціями дозволяє змінювати елементи РКС, і навіть проводити цих елементах оновлення. Як така система служить систему управління конфігураціями Ansible, написана мовою Python. Ansible фокусується на оптимізації та швидкості, і не вимагає установки агентів на керовані вузли - всі функції виконуються за SSH, принцип роботи показаний на рис. 2.3.

Спочатку Ansible була системою управління конфігураціями, що створюється для управління серверною інфраструктурою, проте, в процесі розвитку до системи були додані модулі для роботи з мережевим обладнанням різних виробників.

На жаль, у порівнянні з серверами, функціонал Ansible для роботи з мережевим обладнанням є досить скромним, але виконання базових операцій підтримується. Наприклад, одним із принципів системи Ansible є його іммутабельність, тобто якщо на сервері встановлений бажаний пакет і адміністратор повторить запуск завдання на установку цього пакета, то Ansible це розпізнає і завдання виконувати не буде. Це може бути застосовано як і для конфігурацій мережевого обладнання, але такий функціонал поки не реалізований.

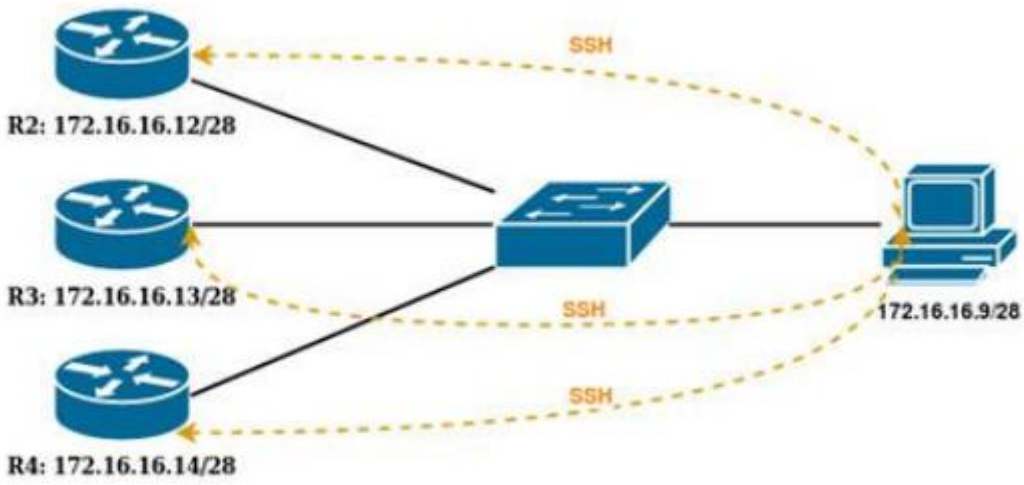

Рис. 2.3. Принцип роботи системи управління конфігураціями Ansible

Скрипти Ansible підтримують два режими роботи: ad-hoc команди та playbook. Ad-hoc команди призначені для виконання разової дії з пристроями або групою пристроїв і виконуються з командного рядка. Playbook є файлом сценарію, що складається з декількох завдань, що виконуються послідовно. Для кожного із завдань може бути визначений набір параметрів і завдання можуть виконуватись залежно, наприклад, якщо перша задача виконалася з помилкою, то наступні завдання не виконуватимуться.

Незалежно від режиму роботи виконання завдання викликається з модуля. Набір модулів встановлюється разом з Ansible і ділиться на дві частини: робота базових модулів гарантована при розвитку системи та виході нових версій продукту.

Алгоритм роботи Ansible виглядає так:

– адміністратор запускає завдання до Ansible;

– Ansible виконує паралельне підключення до списку пристроїв, зазначених у завданні. Число паралельних сесій за умовчанням дорівнює 5, підключення виконується за протоколом SSH, підтримується з'єднання через логін/пароль та сертифікат;

– Ansible виконує збирання інформації про пристрій;

– Ansible виконує завдання;

– якщо завдань виконання більше немає, то Ansible відключається від пристрою, якщо є — виконуються наступні завдання.

Усі перелічені системи пов'язані з так званою експертною системою. Її призначення у загальному випадку - оперування знаннями у певній предметній області для формування рекомендацій або вирішення проблем. У разі автоматизації обслуговуванн розподілених РКС КС експертна система отримує накопичені дані від системи моніторингу, обробляє дані моніторингу за допомогою методів машинного навчання, а потім прогнозує та оповіщає відповідний підрозділ або адміністратора РКС про майбутні несправності відповідно до інвентарної інформації про технічні пристрої та конфігурацією цього пристрою.

2.6. Алгоритми та методи машинного навчання як інструмент технічної діагностики

Методи машинного навчання активно використовуються в різних сферах діяльності: розпізнавання мови, обробці медіафайлів, кредитному скорингу та ін. З точки зору технічного обслуговування один з варіантів застосування методів машинного навчання є застосування задачі бінарної класифікації, яка полягає в наступному: за заданим вектором параметрів технічного об'єкта потрібно визначити, до якого стану (справного чи несправного) відноситься об'єкт.

Обробка інформації та діагностика стану технічного об'єкта проводиться за результатами контролю непрямих показників функціонування об'єкта у процесі експлуатації. При цьому завжди є ризик помилкової тривоги, наприклад, коли справний об'єкт буде визнаний несправним, або навпаки, коли несправність об'єктів не виявлено.

Як вихідні дані розглядаються результати оцінки стану об'єкта. Існує деяка невідома залежність між показниками функціонування об'єкта та його станами. На основі вибірки вихідних даних потрібно відновити залежність, тобто побудувати алгоритм, здатний для заданого набору показників функціонування об'єкта видати досить точну відповідь про його стан.

Вихідні дані для діагностики стану вузла РКС подаються у вигляді матриці X показників функціонування системи, елементи якої X_{ij} - результат i-го спостереження за j-м показником; $i = 1, l,] = 1, ..., p$ (l - кількість рядків, або кількість спостережень, p - кількість стовпців, або кількість показників), і вектор-стовпець відповідей Y, що складається з одиниць (для тих дослідів, в яких об'єкт справний) та нулів при несправному об'єкті. Рядку x_i матриці показників X відповідає певне значення y_i вектора Y, що характеризує справність об'єкта. Множина пар (x_i, y_i) дає вибірку вихідних даних - прецедентів.

Завдання полягає в побудові моделі $a(x, w)$, яка передбачить відповідь Y для будь-якого заданого X. Зазвичай використовуються лінійні моделі [32]:

$$a(x, w) = w_0 + w_l x_l + \cdots + w_p x_p,$$

(2.1)

де $w = (w_0, w_1, \ldots, w_p)$ - вектор параметрів моделі.

При бінарній класифікації замість нуля та одиниці часто використовують множину відповідей $Y = \{-1; +1\}$. У цьому випадку модель алгоритму набуде вигляду:

$$a(x, w) = sign \sum_{j=0}^{p} w_j x_j \qquad (x_0 = 1) \qquad (2.2)$$

Процес підбору оптимальних параметрів алгоритму w_i за вихідними даними називається навчанням алгоритму. Знайдені параметри мають забезпечити оптимальне значення якості функціоналу. У розглянутій задачі найчастіше мінімізується функціонал помилок - це середня кількість неспівпадінь, де $L(a,x_i)$ називають функцією втрат:

$$Q(a, X) = \frac{1}{l} \sum_{i=1}^{l} L(a, x_i) = \frac{1}{l} \sum_{i=1}^{l} [a(x_i) - y_i] \to min \qquad (2.3)$$

Алгоритм, який мінімізує функціонал (2.3), може забезпечувати хороше прогнозування справності об'єкта. Ситуація, коли якість роботи алгоритму на нових об'єктах значно гірша, ніж на вихідній вибірці, свідчить про перенавчання: алгоритм занадто добре підігнаний під навчальну вибірку і не здатний до узагальнення на інші вибірки. Таким чином, побудований алгоритм не зможе передбачати стан об'єкта, що досліджується, при нових параметрах функціонування.

Оцінка якості моделі з погляду можливості прогнозування проводиться за контрольною вибіркою. З цією метою вихідна вибірка з l дослідів поділяється на дві підмножини, які не пересікаються: навчальну вибірку об'єму l_o, за допомогою

якої і вирішується завдання навчання, і контрольну (або тестову) об'єму $l_к = l - l_o$, що не використовується для навчання.

Часто застосовується крос-валідація: вибірка розбивається на N частин (практично часто приймають $N = 5$ чи $N = 10$). Частини (N- 1) використовується для навчання, а решта - для контролю. Послідовно перебираються усі варіанти.

Для кожного варіанта розбиття за вибіркою обсягом l_0 проводиться навчання і визначається функція помилок $Q(a,X)$ на контрольній вибірці l_k. Середнє значення цієї функції з усіх варіантів характеризує узагальнюючу здатність алгоритму.

Методи машинного навчання у час широко розвиваються і активно застосовують у різних галузях діяльності. Використовується багато різних підходів до класифікації. Це і класичні статистичні методи (наївний байєсівський класифікатор, дискримінантний аналіз, логістична регресія, метод опорних векторів) та методи, спеціально орієнтовані на машинне навчання (нейронні мережі), композиційні методи (бегінг, бустинг) та інші.

Наприклад, у статті [33] байєсівський класифікатор використовується для визначення тематики тексту. Були отримані результати, якими можна стверджувати, що обраний автором метод впорався з поставленим завданням. Однак немає доказів того, що цей метод перевершує за результатами інші методи та моделі.

Ансамблеві (композиційні) методи можуть бути використані для прогнозування відтоку клієнтів. Так у статті [34] проводиться аналіз ефективності методів бустингу та бегінга для цих цілей.

Проблема полягає в тому, що не можна заздалегідь визначити, який із вибраних методів забезпечить коректне вирішення задачі, тому часто використовується множина різних методів або їх комбінації, а рішення про застосування приймається за результатами дослідження якості функціоналу для контрольної вибірки.

Іноді будують різні композиції з алгоритмів. При цьому похибки окремих алгоритмів можуть частково компенсуватися. Композиція чи ансамбль

алгоритмів, тобто набір моделей, що використовуються вирішення однієї й тієї завдання, дозволяє підвищити точність класифікації.

Два основних методи побудови композиції - бустинг і бегінг - дають зазвичай точніший результат, ніж застосування окремого алгоритму на конкретному наборі даних.

Бустинг один із ефективних методів машинного навчання. Підсумкове правило у бустингу визначається шляхом виваженого голосування композиції базових правил. При цьому використовується інформація про помилки попередніх правил: ваги об'єктів вибирають таким чином, щоб нове правило точніше працювало на тих об'єктах, на яких з попередніми правилами частіше виникали помилки.

Бегінг використовують за наявності невеликих навчальних вибірок: з наявної вибірки вихідних даних випадково з поверненням формується кілька підмножин такого ж обсягу, як і вихідна вибірка. За підсумками кожного підмножини будується класифікатор, а результати комбінуються шляхом голосування чи усереднення. Часто як базові алгоритми використовуються дерева рішень.

2.7. Висновки до розділу

У другому розділі було вивчено загальні принципи ефективного усунення несправностей. Також розглянуто методи розв'язання задач автоматизації обслуговування РКС.

Запропоновано модель взаємодії технічних систем моніторингу, інвентаризації та системи управління конфігурацій з експертною системою прогнозування та усунення аномалій. Вивчено використання методів машинного навчання у вирішенні задач технічної діагностики.

.

РОЗДІЛ 3
ПРОГНОЗУВАННЯ АНОМАЛІЙ ЗА ДОПОМОГОЮ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ

3.1. Характеристика системних журналів syslog

Мережеві пристрої та сервери час від часу повідомляють про стан апаратного та ПЗ або аномальну подію в повідомленні системного журналу syslog. Приклади таких умов або подій включають зміни стану інтерфейсів, каналів або сусідніх пристроїв (наприклад, стан інтерфейсу змінюється з ввімкненого на вимкнене), експлуатаційне обслуговування (наприклад, вхід або вихід в операційну систему технічного обладнання), оповіщення про стан апаратного забезпечення (наприклад, висока температура процесора) і т. д. Для збору системних журналів виділені сервери розгортаються в інфраструктурі РКС для збору повідомлень системного журналу від множини технологічного обладнання, схема взаємодії обладнання та сервера збору системних журналів представлена на рис. 3.1.

Рис. 3.1. Збір та обробка системних журналів syslog

Як видно з декількох прикладів повідомлень системного журналу, показаних у таблиці 3.1, таке повідомлення, зазвичай, має просту структуру, що

складається з декількох полів, включаючи запис мітки часу, коли пристрій генерував повідомлення системного журналу, ідентифікатор пристрою, який генерував повідомлення, тип повідомлення, що описує короткі характеристики повідомлення, і тіло повідомлення, відображає подробиці події.

Таблиця 3.1

Приклади повідомлень системних журналів

Виробник	Тимчасовий штамп	Ідентифікатор	Тип повідомлення	Коротка характеристика
Виробник 1	Jan 23 14:24:41 2024	Сервер 1	SIF	Interface te-1/1/8, changed state to up (Включено мережний інтерфейс te-1/1/8)
Виробник 1	Mar 19 15:04:11 2024	Комутатор 1	OSPF	A single neighbour should be configured (Необхідно налаштувати сусідній пристрій)
Виробник 2	Apr 21 14:53:05 2024	Сервер 13	10DEVM/2/ POWER FAILED	Power PowerSupply1 failed (відмова гнізда живлення PowerSupply!)
Виробник 2	Sep 11 08:12:32 2024	Комутатор 10	10IFNET/3/ LINK UPDOWN	GigabitEthernet1/0/18 link status is DOWN (Відключено мережний інтерфейс GigabitEthernet - 1/0/18)
Виробник 2	Nov 01 07:29:06 2024	Сервер 5	10CFM/5/CFM SAVE CONFIG SUCCESSFUL LY	Configuration is saved successful (Конфгурація успішно збережена)

Повідомлення syslog є багатим джерелом інформації як діагностики, так попереджуючого прогнозування відмов устаткування у РКС. Однак, така інформація може бути ефективно вилучена лише за допомогою належної обробки журналів, наприклад, з використанням відповідних методів машинного навчання.

Дослідження проводилося на обробці алгоритмом датасета системних

журналів syslog, які були зібрані з 2223 комутаторів, розгорнутих більш ніж у 10 гермозонах, що належать постачальнику інтернет послуг, упродовж двох років.

Збій комутатора відбувається, коли функція, яку надає комутатор (переадресація трафіку), відхиляється від норми. несправністю. Відмови комутатора можна умовно поділити на кілька різних типів:

– зовнішні проблеми, такі як відключення електроживлення;

– проблеми конфігурації обладнання, такі як помилки VPN-тунелювання;

– апаратні збої, наприклад збій, викликаний апаратними помилками, мережевої карти або всього комутатора;

– збій ПЗ через помилки.

Зовнішні проблеми та проблеми з конфігурацією генеруються операторами або іншими пристроями ззовні і, отже, не можуть бути передбачені використанням даних системного журналу Комутатори зазвичай можуть автоматично відновлюватися після збоїв ПЗ, зазвичай, за допомогою перезавантаження, тому в контексті даного дослідження прогнозування збоїв ПЗ на комутаторах не потрібно. Необхідно зосередити увагу на апаратних збоях, які переважають у всіх типах збоїв комутатора (приблизно 33% від усіх збоїв) та загального часу простою (приблизно 72% від усіх збоїв) [36]. Відомо, що збої комутатора значно впливають на продуктивність мереж РКС та центрів обробки даних. Наприклад, збої комутатора для сервісу Hosting.com [37] призвели до збоїв у їхній РКС.

3.2. Модель прогнозування відмови

На рис. 3.2 показана задача прогнозування відмови комутатора. Для цієї задачі (безперервний) час дискретизується на невеликі часові інтервали фКСованої тривалості (скажімо, 15 хвилин) і нумерується як цілі числа.

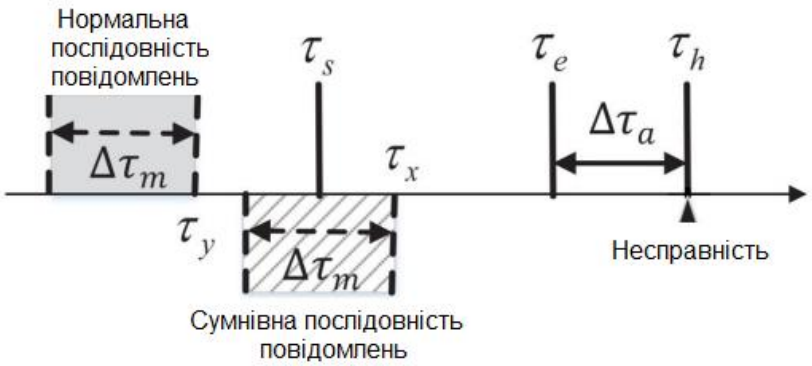

Рис. 3.2. Модель прогнозування відмови

Припустимо, що збій відбувається при τ_h. Завдання прогнозування відмови полягає у прогнозуванні відмови комутатора у будь-який момент часу τ_x у межах [τ_s, τ_e]. Оскільки операторам мережі потрібно не більше $\Delta\tau_a$ часу , щоб відреагувати на позитивне передбачення несправності, наприклад, для перенаправлення трафіку або заміни комутатора, що виявляє збій τ_e випереджає τ_h на $\Delta\tau_a$. Збільшення інтервалу часу [τ_s, τ_e] завжди призводить до підвищення точності прогнозування відмов (наприклад, якщо тривалість інтервалу дорівнюватиме нескінченності, то позитивний прогноз відмов гарантовано буде правильним), якщо ж тривалість [τ_s, τ_e] занадто велика, то прогноз (позитивна відмова) малопридатний, оскільки він не дає чіткого керівництва операторам щодо необхідності здійснення необхідних дій.

3.3. Створення шаблонів повідомлень системних журналів

В табл. 3.2 є набір повідомлень системного журналу, які відносяться до типу повідомлень SIF і описувати зміни стану комутатора, докладні повідомлення можуть бути зовсім різними. Поле докладних повідомлень системного журналу по суті є текстом у вільній формі, і операційна система комутатора, зазвичай, генерує це поле, використовуючи функцію «printf» з докладною інформацією як змінні, такі як місце розташування (мережева карта / порт / інтерфейс), коефіцієнт втрати пакетів, IP -адреса і т.д.

Наприклад, поле докладного повідомлення системного журналу в першому

рядку табл. 3.2, *" Interface ae3, changed state to down»,* означає, що подія вплинула на інтерфейс *ae3,* і, як наслідок, стан інтерфейс змінився на неактивний. Іншими словами, *ae3* є змінною докладної інформації, тоді як решта тексту, тобто. *"Interface ..., changed state to down*», є шаблонним виведенням операційної системи комутатора і можуть використовуватися як підтип для повідомлень системного журналу. які належать цього типу повідомлення, тобто SIF.

Таблиця 3.2

Приклад послідовності повідомлень перед появою несправності в комутаторі

Ідентифікатор повідомлення	Тимчасовий штамп	Тип повідомлен	Коротка характеристика
D1	9:21:10	SIF	Interface ae3, changed state to down (Відключено мережний інтерфейс ae3)
D2	9:22:23	SIF	Vlan-interface vlan22, changed state to down (Відключено vlan –інтерфейс vlan22)
D3	9:23:45	SIF	Interface ae3, changed state to up (Включено мережний інтерфейс ae3)
D4	9:25:28	SIF	Vlan-interface vlan22, changed state to up (Включено vlan -інтерфейс vlan22)
D5	9:27:20	OSPF	Neighbour(id:10.231.0.40, addr:10.231.38.85) on vlan22, changed state full to down (Стан сусіда OSPF (Rid:10.231.0.40, addr: 10.231.38.85) у vlan22 перейшов в неактиний)
D6	9:30:31	SIF	Interface ae1, changed state to Down (Відключено мережний інтерфейс ae1)
D7	9:32:34	SIF	Vlan-interface vlan20, changed state to down (Відключено vlan –інтерфейс vlan20)
D8	9:37:21	SIF	Interface ae1, changed state to up (Включено мережний інтерфейс ae1)
D9	9:38:24	OSPF	Neighbour(rid:10.231.0.40, addr:10.231.36.85) on vlan20,

Коли ми замінюємо змінну для докладного поля повідомлення з повідомлень системного журналу, тобто номери інтерфейсу або номери інтерфейсу vlan, використовуючи той самий символ, наприклад, зірку (*), як показано у табл. 3.3, існує лише чотири різні структури чи підтипу повідомлень системного журналу.

Таблиця 3.3

Приклад послідовності повідомлень перед появою несправності в комутаторі

№	Структура підтипу	Відповідний ідентифікатор повідомлення
1	Interface *, changed state todown	D1, D6, D13
2	Interface *, changed state to up	D3, D8, D15
3	Vlan-interface *, changed state to down	D2, D7, D14
4	Vlan-interface *, changed state to up	D4, D10, D16

Нехай $I = a_1 , a_2 , ..., a_m$ буде набором окремих слів, які зустрічаються в наборі повідомлень $DM = \{M_1, M_2, ..., M_n \}$, де кожне M є повідомленням системного журналу.

Частота появи (тобто скільки разів зустрічається слово у вибірці повідомлень) словосполучення A є кількість окремих повідомлень, що містять A в DM. Причому A вважається шаблоном, якщо зустрічається часто. У другому стовпці табл. 3.4 показаний приклад набору повідомлень системного журналу $DM = M_1, M_2 , ..., M_8$, в якому кожне повідомлення відноситься до типу повідомлення «SIF».

Таблиця 3.4

Повідомлення, що відносяться до типу повідомлення «SIF» у табл. 3.2, та слова, упорядковані відповідно до *L*

№ повідомлення	Коротка характеристика	Слова збудовані згідно частоти появи
M_1	Interface ae3, changed state to down	«changed», «state», «to», «Interface», «down», «ae3»
M_2	Vlan-interface vlan22, changed state to down	«changed», «state», «to», «Vlan-interface», «down», «vlan22»
M_3	Interface ae3, changed state to up	«changed», «state», «to», «Interface», «up», «ae3»
M_4	Vlan-interface vlan22, changed state to up	«changed», «state», «to», «Vlan-interface», «up», «vlan22»
M_5	Interface ael, changed state to down	«changed», «state», «to», «Interface», «down», «ael»
M_6	Vlan-interface vlan20, changed state to down	«changed», «state», «to», «Vlan-interface», «down», «vlan20»
M_7	Interface ael, changed state to up	«changed», «state», «to», «Interface», «up», «ael»
M_8	Vlan-interface vlan20, changed state to up	«changed», «state», «to», «Vlan-interface», «up», «vlan20»

Дерево рішень при цьому *DM* побудовано з допомогою алгоритму, представленого на рис. 3.3.

По-перше, алгоритм побудови дерева сканує *DM* один раз і виводить список *L* слів у порядку зменшення їх частот появи (число після кожного «:»). Виходячи з цього отримаємо, L = {(« *changed* «: 8), (« *state* «: 8), («to»:8), (« *Interface»:4*), *(«Vlan-interface»:4),* («*down*»:4), («*up*»:4), («*ae3*»:2), («*aei*»:2), («*vlan22*»:2), («*vlan20*»:2)}.

Далі створюється корінь дерева, позначеного типом повідомлення, який у даному випадку є «SIF». Алгоритм побудови дерева шаблон сканує *DM* вдруге. Обробка M_1 призводить до побудови першого шляху або гілки дерева: {*«changed», «state», «to», «Interface», «down»,* «ае3»}. Причому ці слова впорядковані відповідно до порядку слів у *L*. Коли M_2 обробляється, тому що його впорядкований список слів *{changed, state, to, Vlan-interface, down, vlan22 }* має

спільні префКСи *{«changed», «state»,* «to»} з існуючим шляхом/гілкою *{«changed», «state», «to», «Interface», «down»,* «ае3»}, то нова гілка *{« Vlan-interface», "down», "vlan22"}* створюється виходячи з вузла "to".

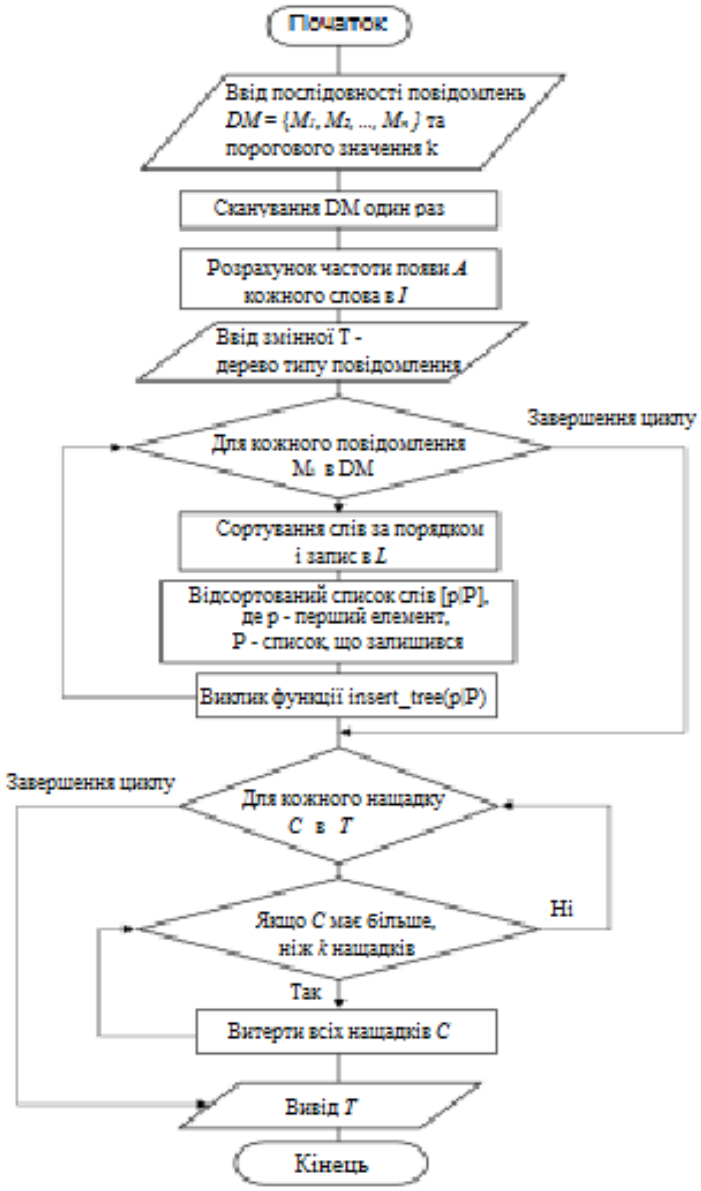

Рис. 3.3. Блок-схема алгоритму побудови дерев шаблонів повідомлень

Інші шість повідомлень з табл. 3.4 обробляються аналогічним чином, у результаті виходить підсумкове дерево, показане на рис. 3.4. У ньому видно, що корінь позначений типом повідомлення, кожен шлях від кореня до листа відповідає шаблону повідомлення. Кожен некореневий вузол у дереві має лише один атрибут, і проіменований словом із повідомлень певного типу системного

журналу, що представляє цей вузол.

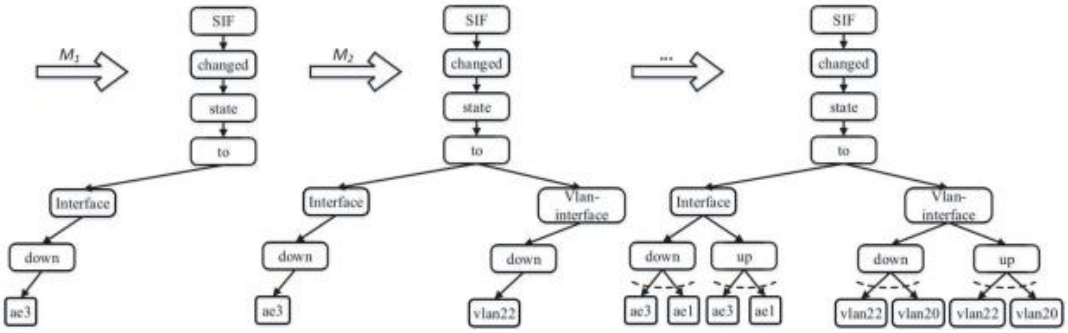

Рис. 3.4. Приклад побудови дерева шаблону повідомлень типу «SIF»

З алгоритму також зрозуміло, що такі вставки можуть виконуватися поступово шляхом сканування лише нещодавно отриманих повідомлень системного журналу (після оновлення) двічі. Більш конкретно, коли приходить нове повідомлення M_{new}, яке не відповідає жодному шаблону, нова гілка вставляється в дерево шляхом виклику функції insert_tree () в алгоритмі. Якщо одне або кілька слів у M_{new} не існує в L, ми додамо ці слова до хвоста L. Коли комутатор починає генерувати нові підтипи повідомлень, ймовірно, що в певний момент ці повідомлення будуть містити достатню кількість (більше, ніж поріг скорочення k) різних слів параметрів (наприклад, таких як *«ae3», «vlan22»* і т.д. .) як дочірніх елементів вузла, тому цей вузол сам стає листом після відсікання.

Процес поступового навчання показаний на рис. 3.5.

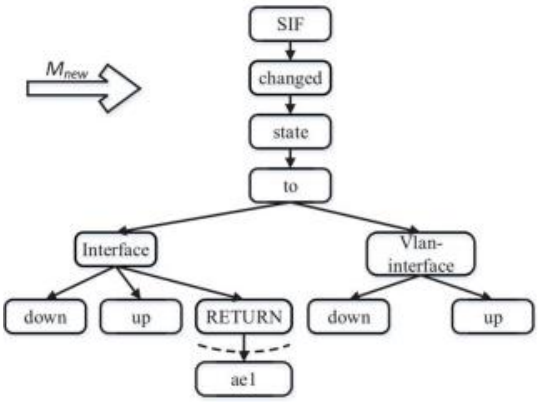

Рис. 3.5. Приклад дерева шаблону повідомлень після появи повідомлення нового типу

Припустимо, що надходить нове повідомлення M_{new} = «*Interface ae1 changed state to RETURN»,* і що перед цим дерево має вигляд найправішого дерева з рис. 3.3 без кінцевих вузлів («*ae3», «ae1»,* «*vlan22»,* і т.д.).

Поява нового повідомлення M_{new} приводить до вставки гілки *(«RETURN»* → «*ae1*») у дерево шаблону, і останній вузол «*ae1*» зрештою буде відтятий, як пояснюється раніше.

Вивчивши шаблони із повідомлень системного журналу, ми можемо зіставити історичне повідомлення системного журналу або нещодавнє повідомлення системного журналу з конкретним шаблоном, закодованим як числове значення. Більш конкретно, для послідовності повідомлень системного журналу $(s_1, s_2, ..., s_n)$, для кожного повідомлення s_i підбирається відповідний шаблон повідомлення t_{si}, ми називаємо $(t_{s1}, t_{s2}, ..., t_{sn})$ як послідовність шаблонів повідомлень $(s_1, s_2 ..., s_n)$.

3.4. Навчання алгоритму та прогнозування

Метод випадкового лісу використовує кілька двійкових дерев ухвалення рішень (тобто ліс) на етапі навчання. У кожному двійковому дереві рішень клас (у разі або 1 - сумнівна послідовність, або 0 - нормальна послідовність), з допомогою голосування класифікаторів, визначених окремими деревами (за 1 чи 0). Потім серед голосів кандидата виявляється більшість голосів, і результат більшості береться за остаточне рішення. Цей алгоритм написаний у вигляді програми мовою програмування Python.

У навчальному компоненті кожен сумнівний момент часу позначається 1 (позитивно), і аналогічно кожен нормальний момент позначається 0 (негативно). Таким чином, ми будуємо вектор мітки v для $(\tau_1, \tau_2, ..., \tau_м)$. Потім у модель випадкового лісу вводиться матриця A як вибірки і v як міток, модель проходить навчання і на виводі виходять параметри, що використовуються для передбачення збоїв. У компоненті передбачення, коли генерується відповідна послідовність повідомлень у певний момент часу модель перетворює повідомлення syslog

шаблони, витягує ознаки з послідовностей шаблонів і визначає, чи є послідовність шаблонів індикатором несправності, на основі параметрів, записаних у процесі навчання. Для наочності вивчені шаблони записуються у текстовий файл. Причому порядок слів у шаблонах побудований у порядку зменшення їх частот появи.

Як метод машинного навчання було обрано алгоритм випадкового лісу, а не метод опорних векторів з двох причин [39].

Для проведення дослідження постачальником інтернет- послуг було надано набір даних комутаторів з найбільшою кількістю відмов апаратного забезпечення від одного постачальника, а також проаналізовано всі апаратні збої та системні журнали комутаторів за дворічний період.

У табл. 3.5 наведено дані про кількість виявлених несправностей апаратури, кількість несправних комутаторів, загальну кількість комутаторів, кількість сумнівних моментів часу та кількість нормальних моментів часу.

Таблиця 3.5

Детальна інформація щодо дослідженого датасету

Параметр	Кількість
Виявлені несправності	228
Несправні комутатори	131
Загальна кількість комутаторів	2223
Сумнівні моменти часу	1273
Нормальні моменти часу	5 516 435

Історична інформація щодо несправностей у роботі комутаторів збиралася з трьох джерел:

– проблемні квитки при виявленні аномалії РКС;

– проактивне виявлення відмови комутатора за допомогою SNMP-опитування;

– проактивне виявлення збою комутатора через системні журнали syslog.

Щоб змоделювати процедуру онлайн-прогнозування в експерименті ми щоразу змінюємо $\Delta\tau_m$ і отримуємо сумнівні та нормальні моменти часу відповідно до визначень, наведених у п. 3.3. Для експерименту було встановлено такі параметри $\tau_e - \tau_s$ = 24 год, $\Delta\tau_m$= 2 год, $\Delta\tau_a$ = 30 хв, θ = 5 і ξ = 15 хв. Вибір даних параметрів обумовлений в такий спосіб. На основі емпіричного досвіду мережні оператори вважають, що дані системних журналів за останні 24 години до збою є індикатором несправності, і, таким чином, можуть бути використані для прогнозування несправностей. Після аналізу системних журналів до десятків відмов було виявлено, що в більшості випадків (більше 90%), системні журнали за будь-які дві години в системних журналах, які знаходяться в межах 24 годин до відмови, можуть мати сумнівний шаблон. Крім того, моніторинг системних журналів за 24-годинний період для процедури онлайн прогнозування є надто інтенсивним з обчислювальної точки зору.

Більше того, якщо у відповідній послідовності повідомлень моменту часу менше п'яти повідомлень у системному журналі, послідовність повідомлень буде занадто короткою, щоб визначити сумнівний шаблон, і тому не може бути використаний для прогнозування відмови. Час реакції оператора на позитивний прогноз відмови, тобто $\Delta\tau_a$**,** слідує тривалість кожного моменту часу ξ.. Так як $\Delta\tau_a$ = 30 хв, ξ може бути 30, 15, 10, 5, 2, 1. Якщо значення ξ надто мале, тобто ξ , ξ может быть 30, 15, 10, 5, 2, 1. Если значение ξ слишком мало, т.е. $\xi \in$ {1 хв, 2 хв, 5 хв, 10 хв}, то буде занадто мало моментів часу, що містять більше θ = 5 повідомлень системних журналів. З іншого боку, якщо ξ буде занадто великим, наприклад, ξ = 30 хв, кількість сумнівних моментів часу скоротиться більш ніж на 50% і, таким чином, сильно вплине на навчання моделі. Отже, у дослідженні встановлюється ξ 15 хв.

За підсумками прогнозування за допомогою моделі випадкового лісу, написаної мовою Python, були отримані результати, описані в табл. 3.6. Результати досліджень показують, що серед всього обсягу даних повідомлень системних журналів були передбачені моменти часу, що містять послідовності повідомлень, які призведуть до відмови комутатора в наступні 24 години, серед

яких є помилкові прогнози сумнівних моментів часу

Таблиця 3.6

Результати прогнозування

Позитивно виявлені сумнівні моменти часу	Помилково виявлені сумнівні моменти часу	Пропущені сумнівні моменти часу	Фактичне кількість сумнівних моментів часу	Нормальні моменти часу
947	137	326	1273	5 516 435

3.5. Оцінка якості моделі прогнозування

Працюючи з алгоритмами машинного навчання з метою оцінки якості моделей використовуються показники чи метрики, які вибір і аналіз — неодмінна частина роботи у дослідженні.

Зазвичай проводиться оцінка чотирма метриками, які мають інтуїтивно зрозумілі інтерпретації: точність *(precision),* повнота *(recall).,* F- міра *(-Fmeasure)* і хибнопозитивне співвідношення *(FPR).*

У дослідженні є два класи - сумнівний момент часу, що позначається 1, і нормальний момент часу, що позначається 0. Є алгоритм, що передбачає належність кожного об'єкта одному з класів, тоді матриця помилок класифікації виглядатиме таким чином, показаним у табл. 3.7.

Таблиця 3.7

Матриця помилок

Прогноз	Реальність	
	1	0
1	TP (True Positive) - істинно-позитивне рішення	FP (False Positive) - Хибнопозитивне рішення
0	FN (False Negative) - Хибнонегативне рішення	TN (True Negative) - істинно негативне рішення

У таблиці 3.8 виведено результат передбачень моделі з погляду матриці помилок.

Таблиця 3.8

Результати прогнозу з погляду матриці помилок

Прогноз	Реальність	
	1	0
1	TP = 947	FP = 137
0	FN = 326	TN = 5 516 298

Показники оцінки точності прогнозу представлені у табл. 3.9.

Таблиця 3.9

Показники точності, повноти, F-міра та хибнопозитивного співвідношення

Точність	Повнота	F-міра	FPR
87,35 %	74,36 %	80,33 %	$2{,}49 * 10^{-5}$

На рис. 3.6 побудовано криву точність-повноту, яка показує якість моделі прогнозування

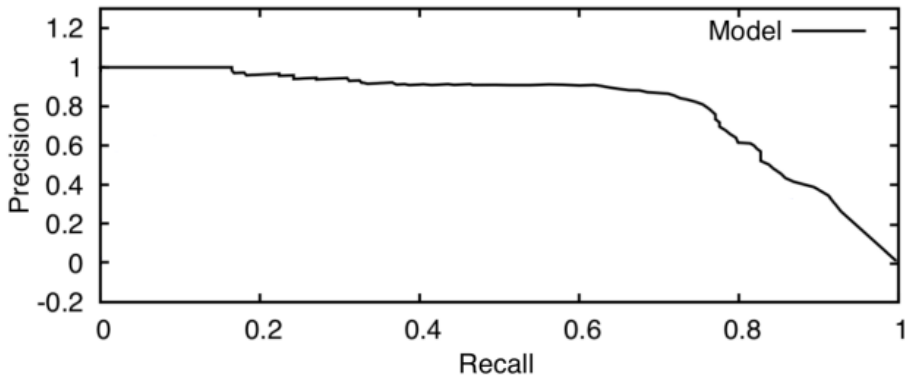

Рис. 3.6. Крива точність-повнота

Чим більше крива прагне до 1 (100% точність і повнота), тим точніше

працює модель машинного навчання і означає хорошу ефективність, в той час як прагнення кривої в нижній лівий кут означає погану продуктивність.

Щоб оцінити вплив виявлення ознак серійності та частоти на прогнозування, скористаємося цією ж кривою для оцінки результатів прогнозування, але в порівнянні вихідної кривої з кривою, побудованої за результатами без виявлення ознак серійності та з кривою, побудованої за результатами без виявлення ознак частоти, результат наочно видно на рис. 3.7.

Крива чорного кольору є модель прогнозування з виявленням ознак, червона пунктирна крива представляє модель без виявлення ознак серійності, а синя штрих-пунктирна крива відображає модель без виявлення ознаки частоти. За графіком кривих видно, що з добуванні однієї з функції виявлення ознак, модель прогнозування працює негаразд ефективно, як раніше. Особливо це видно за результатами без виявлення ознак серійності.

На підставі цього можна зробити висновок, що виявлення ознак серійності та частоти важливі для прогнозування несправностей та збоїв, і що метод дворазового виявлення найбільшої загальної підпослідовності успішно виявляє ознаки серійності послідовності шаблонів повідомлень для прогнозування збоїв.

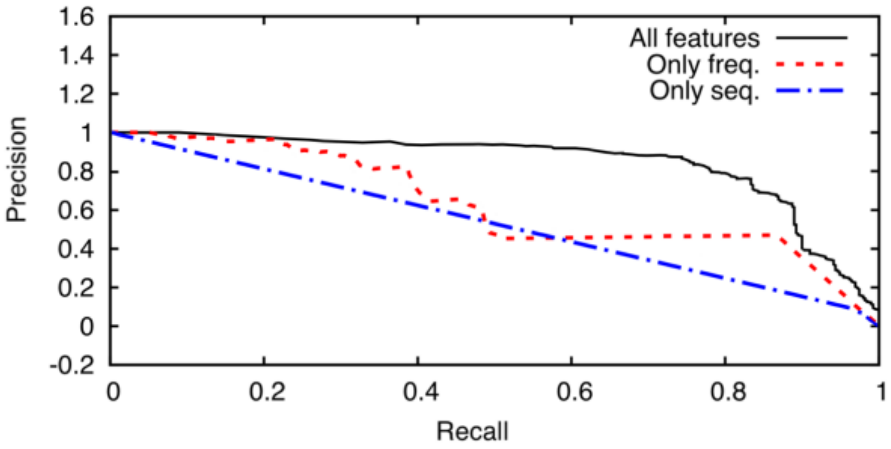

Рис. 3.7. Порівняння кривих точність-повнота

Таким чином, при прогнозуванні відмов комутаторів з датасету системних журналів за допомогою методів машинного навчання, а саме алгоритму

випадкових лісів, точність прогнозів склала 87,35%, повнота прогнозу склала 74,36%, F-міра склала 80,33% і показник хибно-позитивного співвідношення становить 2,49-10 $^{-5}$, що дозволяє говорити про «хорошу» точність прогнозу.

3.6. Висновок до розділу

У цьому розділі було проведено прогнозування майбутніх помилок обладнання КС з прикладу прогнозування несправностей комутаторів.

Вибір помилок комутаторів як дослідження моделі прогнозування збоїв обумовлений тим, що на практиці серед усіх відмов пристроїв в інфраструктурі РКС збої в роботі комутаторів є домінуючим типом як за часом простою, так і за кількістю випадків. Несправності комутатора, якщо їх не усунути терміново, можуть призвести до зниження продуктивності РКС.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Охорона праці

Метою кваліфікаційної роботи магістра є експериментальне дослідження файлів системних журналів, на основі яких проводиться тренування алгоритму машинного навчання з метою прогнозування майбутніх несправностей пристроїв РКС (на прикладі системних журналів комутаторів). Оскільки, проведення такого роду робіт передбачає застосування комп'ютерної техніки, зокрема ПК та периферійних пристроїв, то обов'язковим є дотримання вимог з охорони праці і техніки безпеки.

Для ефективної і безпечної роботи колективу працівників з експериментального дослідження файлів системних журналів, на основі яких проводиться тренування алгоритму машинного навчання з метою прогнозування майбутніх несправностей пристроїв РКС (на прикладі системних журналів комутаторів), необхідно організувати безпечні умови праці. При цьому керівник організації несе безпосередню відповідальність за порушення нормативно-правових актів з охорони праці [40]. Окрім цього, на робочих місцях працівників необхідно забезпечити дотримання вимог, затверджених Наказом Мінсоцполітики від 14.02.2018 за № 207 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями». Згідно Вимог приміщення, де розміщені робочі місця операторів, крім приміщень, у яких розміщені робочі місця операторів великих ЕОМ загального призначення (сервер), мають бути оснащені системою автоматичної пожежної сигналізації відповідно до цих вимог;

– переліку однотипних за призначенням об'єктів, які підлягають обладнанню автоматичними установками пожежогасіння та пожежної сигналізації, затвердженого наказом Міністерства України з питань надзвичайних ситуацій та у справах захисту населення від наслідків Чорнобильської катастрофи

від 22.08.2005 N 161, зареєстрованого в Міністерстві юстиції України 05.09.2005 за N 990/11270 (НАПБ Б.06.004-2005);

– Державних будівельних норм «Інженерне обладнання будинків і споруд. Пожежна автоматика будинків і споруд», затверджених наказом Держбуду України від 28.10.98 N 247 (далі - ДБН В.2.5-56:2014, з димовими пожежними сповіщувачами та переносними вуглекислотними вогнегасниками.

В інших приміщеннях допускається встановлювати теплові пожежні сповіщувачі. Приміщення, де розміщені робочі місця операторів, мають бути оснащені вогнегасниками, кількість яких визначається згідно з вимогами ДСТУ 4297:2004 «Пожежна техніка. Технічне обслуговування вогнегасників». Загальні технічні вимоги і з урахуванням граничнодопустимих концентрацій вогнегасної рідини відповідно до вимог НАПБ А.01.001-2014. Приміщення, в яких розміщуються робочі місця операторів сервера загального призначення, обладнуються системою автоматичної пожежної сигналізації та засобами пожежогасіння відповідно до вимог ДБН В.2.5-56:2014, НАПБ А.01.001-2014 і вимог нормативно-технічної та експлуатаційної документації виробника. Проходи до засобів пожежогасіння мають бути вільними.

Лінія електромережі для живлення комп’ютера та периферійних пристроїв повинні бути виконаними як окрема групова трипровідна мережа шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Не допускається використовувати нульовий робочий провідник як нульовий захисний провідник. Нульовий захисний провідник прокладається від стійки групового розподільного щита, розподільного пункту до розеток електроживлення. Не допускається підключати на щиті до одного контактного затискача нульовий робочий та нульовий захисний провідники.

Площа перерізу нульового робочого та нульового захисного провідника в груповій трипровідній мережі має бути не менше площі перерізу фазового провідника. Усі провідники мають відповідати номінальним параметрам мережі та навантаження, умовам навколишнього середовища, умовам розподілу

провідників, температурному режиму та типам апаратури захисту, вимогам НПАОП 40.1-1.01-97.

У приміщенні, де одночасно експлуатуються понад п’ять комп’ютерів, на помітному, доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім освітлення. Комп’ютери повинні підключатися до електромережі тільки за допомогою справних штепсельних з'єднань і електророзеток заводського виготовлення.

У штепсельних з'єднаннях та електророзетках, крім контактів фазового та нульового робочого провідників, мають бути спеціальні контакти для підключення нульового захисного провідника. Їхня конструкція має бути такою, щоб приєднання нульового захисного провідника відбувалося раніше, ніж приєднання фазового та нульового робочого провідників. Порядок роз’єднання при відключенні має бути зворотним. Не допускається підключати комп’ютери до звичайної двопровідної електромережі, в тому числі – з використанням перехідних пристроїв. Електромережі штепсельних з'єднань та електророзеток для живлення комп’ютерної техніки повинні бути виконаними за магістральною схемою, по 3-6 з'єднань або електророзеток в одному колі. Штепсельні з'єднання та електророзетки для напруги 12 В та 42 В за своєю конструкцією мають відрізнятися від штепсельних з'єднань для напруги 127 В та 220 В. Штепсельні з'єднання та електророзетки, розраховані на напругу 12 В та 42 В, мають візуально (за кольором) відрізнятися від кольору штепсельних з'єднань, розрахованих на напругу 127 В та 220 В.

При підвищенні ефективності контролю доступу в приміщення, де для забезпечення безпеки мешканців, співробітників і збереження майна використовуються ДС, важливим, з точки зору охорони праці, є забезпечення достатньої величини природного та штучного освітлення, які визначені у НПАОП 0.00-7.15-18. Організація робочого місця фахівця із дослідження методів та програмно-апаратних засобів оптимізаційних процесів на основі ГА повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам ДСТУ 8604:2015 «Дизайн і ергономіка. Робоче місце для

виконання робіт у положенні сидячи. Загальні ергономічні вимоги». Відстань від екрана до ока фахівців, які працюють за комп'ютером визначається згідно з вимогами ДСанПіН 3.3.2.007-98.

Розміщення принтера або іншого пристрою введення-виведення інформації на робочому місці має забезпечувати добру видимість екрана комп'ютера, зручність ручного керування пристроєм введення-виведення інформації в зоні досяжності моторного поля згідно з вимогами ДСанПіН 3.3.2.007-98.

Таким чином, у результаті аналізу вимог щодо охорони праці користувачів комп'ютерів, визначено особливості організації робочих місць, вимог з електробезпеки, природного та штучного освітлення для ефективної і безпечної роботи фахівців з експериментального дослідження файлів системних журналів, на основі яких проводиться тренування алгоритму машинного навчання з метою прогнозування майбутніх несправностей пристроїв РКС.

4.2. Комп'ютерне забезпечення процесу оцінки радіаційної та хімічної обстановки

Екологічне співтовариство розробило сімейство інструментів комплексної екологічної оцінки. ПЗ і послуги (ESS), комерційна група IIASA, включаючи AirWare (для повітряних проблеми якості), WaterWare (для якості води), CityWare (якість повітря і води в контексті великих міст) і EIAxpert (для надання допомоги із загальним впливом на навколишнє середовище). Функціональність в цілому схожа на RAISON, хоча з великим акцентом на моделювання і меншим акцентом на керування даними. Знову ж таки, інструменти ESS розроблені як модульні набори інструментів (доступні спеціальні системи для вирішення конкретних завдань). Компоненти включають стандартні імітаційні моделі, включаючи моделі ISC і PBM Агентства з охорони навколишнього середовища США, управління даними, в тому числі ГІС, аналіз даних (наприклад, аналіз часових рядів даних спостережень), візуалізація, а також оптимізація [41].

Іноді немає готових моделей, придатних для конкретного застосування, але тягар розробки нової програми на Фортрані або C / C ++ є надмірним. Розробка моделі оточення може відносно легко реалізувати власні моделі комп'ютерів і не турбуватися про включення процедур для вирішення рівнянь, візуалізації і т. д. Як правило, за допомогою цих інструментів користувач просто повинен вказати свою модель, використовуючи або математичні рівняння, або спеціальні графічні символи або значки, які безпосередньо представляють поведінку системи.

На даний момент є розроблені моделі комп'ютерного забезпечення процесу для оцінки радіаційної та хімічної обстановки.

GEMS – це система на основі моделей, яка підтримує оцінки схильності і ризику, надаючи доступ до одиночних і мультимедійних моделям експозиції, фізико-хімічні властивості методи оцінки, статистичний аналіз, графічні та картографічні програми з відповідними даними на навколишнє середовище, джерела, рецептори і популяції. У розробці з 1981 року, GEMS надає аналітикам 84 84 інтерактивний, легко досліджуваний інтерфейс для різних моделей, програм і даних, які необхідні для оцінки хімічного впливу і ризику [41].

HSPF – це комплексний пакет для моделювання кількості і якості стоків з багатоцільових водозборів і процесів радіації, що відбуваються в потоках або повністю змішаних озерах. Це дозволяє інтегроване моделювання землі і грунту, процесів забруднення при гідравлічній і осадово-хімічній взаємодії. Результатом моделювання є тимчасові дані витрати стоку, концентрація поживних речовин і пестицидів, а також дані кількості і якості води в будь-якій точці водозбору. Алгоритми якості води включають динаміку BOD / DO, вуглець, азот і фосфор. Процеси трансформації, які включені в модель це: гідроліз, фотоліз, окислення, випаровування, сорбція і биодеградація. Вторинні або «дочірні» хімічні речовини також моделюються.

Вимоги до даних для моделі можуть бути досить широкими в залежності від конкретного застосування.

Модель MMSOILS – це методологія оцінки впливу на людину і ризику для здоров'я, пов'язаних з викидами забруднень з небезпечних відходів.

Мультимедійна модель, що стосується перенесення хімічної речовини в ґрунтові води, поверхневі води, атмосферу і накопичення в їжі. Шляхи впливу на людину, які розглянуті в методології включають: потрапляння в грунт, вдихання летких речовин в повітря і тверді частинки, шкірний контакт, прийом питної води і т.д. Ризик, пов'язаний із загальною дозою опромінення, розраховується на основі хімічної токсичності [41].

4.3. Висновки до розділу

В цьому розділі проаналізовано важливі питання охорони праці та безпеки в надзвичайних ситуаціях, висвітлено питання комп’ютерного забезпечення процесу оцінки радіаційної та хімічної обстановки.

ВИСНОВКИ

Під час роботи над кваліфікаційною роботою було проведено аналіз робіт, присвячених існуючим методам діагностики та усунення несправностей КС. Аналіз робіт показав, що немає узагальненої моделі, що дозволяє вирішувати будь-яку поставлену завдання діагностування чи прогнозування. Процес реалізації моделі повинен базуватись на можливостях обраних класів моделей та вимогах до вирішення поставленого завдання.

У процесі аналізу сформульовано деякі пропозиції щодо використання сукупності систем технічного обліку, моніторингу та системи прогнозування для реалізації концептуальної моделі автоматизації обслуговування РКС.

У ході дослідження було розроблено програму моделі машинного навчання для вирішення задачі прогнозування та усунення аномалій в КС, на основі алгоритму випадкового лісу мовою програмування Python. Було проведено навчання моделі за допомогою набору даних системних журналів syslog комутаторів певного виробника, висновком якого були шаблонні конструкції повідомлень системних журналів комутаторів. Далі прогнозування було зроблено в інтервалі часу рівним 24 годин, використовуючи той самий набір даних у порівнянні з шаблонними конструкціями повідомлень системних журналів з урахуванням ознак серійності та частоти повідомлень.

Результати досліджень були порівняні з історичними даними про відмови та проблемні квитки, і показали, що серед усього обсягу даних повідомлень системних журналів, існували моменти часу, що містять послідовності повідомлень, які передбачають відмову комутатора в наступні 24 години. Працездатність методу продемонстрована метриками оцінки якості моделі машинного навчання. Було досягнуто значення точності, що дорівнює 87,35%, повнота склала 74,36%, F -міри, що дорівнює 80,33% і показник хибнопозитивного співвідношення становить $2{,}49\text{-}10^{-5}$, що дозволяє говорити про «хорошу» точність прогнозу.

Було розглянуто суть алгоритму випадкового лісу та методу дворазового виявлення найбільшої загальної підпослідовності, що лежить в основі знаходження ознак серійності послідовностей шаблонів. Встановлено, що метод дворазового виявлення найбільшої загальної підпослідовності успішно здобуває ознаки серійності послідовності шаблонів повідомлень та покращує точність прогнозу моделі.

Прогнозування несправностей на основі методу машинного навчання випадкового лісу дає можливість завчасно приготуватися до усунення майбутньої несправності, або в деяких випадках до запобігання цим несправностям, що дозволить ефективно здійснювати обслуговування РКС.

Виконаний прогноз про майбутні несправності комутаторів дозволить більш ефективно здійснювати обслуговування РКС на підприємствах зв'язку, що позитивно вплине на безперервність надання телекомунікаційних послуг, дозволить підвищити надійність РКС. Застосування методів прогнозування несправностей у РКС також допоможе уникнути тривалих перебоїв у роботі систем, оскільки знаючи про збій, відповідний підрозділ або фахівець може завчасно вжити необхідних заходів для мінімізації або скасування негативних наслідків несправності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Глобальна інформаційна інфраструктура інфокомунікаційних мереж та систем : навч. посібник / уклад. : Ю.О. Ушенко, А.Л. Негрич, О.В. Галочкін. Чернівці : Чернівецький нац. ун-т ім. Ю. Федьковича, 2021. 224 с.
2. Особливості розподілених систем. URL: https://moodle.znu.edu.ua/pluginfile.php/486881/mod_resource/content/1/Глоба%20книга%20Том1-21-31.pdf (дата звернення:17.11.2024).
3. Луцик Н.С., Луцків А.М., Осухівська Г.М., Тиш Є.В. Програма та методичні рекомендації з проходження практики за тематикою кваліфікаційної роботи для студентів спеціальності 123 «Комп'ютерна інженерія» другого (магістерського) рівня вищої освіти усіх форм навчання. Тернопіль. ТНТУ. 2024. 45 с.
4. Луцик Н. С., Луцків А. М., Осухівська Г. М., Тиш Є. В. Методичні рекомендації до виконання кваліфікаційної роботи магістра для студентів спеціальності 123 «Комп'ютерна інженерія» другого (магістерського) рівня вищої освіти усіх форм навчання. Тернопіль. ТНТУ. 2024. 44 с.
5. Варавін А.В., Лещишин Ю.З., Чайковський А.В. Методичні вказівки до виконання курсового проєкту з дисципліни «Дослідження і проєктування комп'ютерних систем та мереж» для здобувачів другого (магістерського) рівня вищої освіти спеціальності 123 «Комп'ютерна інженерія» усіх форм навчання. Тернопіль. ТНТУ, 2024. 32 с.
6. Микитишин А. Г., Митник М. М., Стухляк П. Д., Пасічник В. В. Комп'ютерні мережі. Книга 1 [навчальний посібник]. Львів : «Магнолія 2006», 2013. 256 с.
7. Микитишин А. Г., Митник М. М., Стухляк П. Д., Пасічник В. В. Комп'ютерні мережі. Книга 2. [навчальний посібник]. Львів : "Магнолія 2006", 2014. 312 с..
8. Погоруй А. О., Чемерис О.А. Вступ до теорії випадкових процесів : навчальний посібник. Житомир : Вид-во ЖДУ ім. І. Франка, 2020. 70 с

9. Croll A. Complete Web Monitoring. – Sean Power: O'Reilly Media, 2009. 672 p.

10. McCarthy K. 10 Things I Learned Deploying Graphite, 2013. URL: https://kevinmccarthy.org/2013/07/18/10-things-i-learned-deploying-graphite/ (дата звертання: 27.04.2019).

11. Сайт NewRelic. URL: http://newrelic.com/servermonitoring (дата звертання: 19.11.2024).

12. Doll B. New Relic Architecture – Collecting 20+ Billion Metrics a Day URL: http://highscalability.com/blog/2011/7/18/new-relicarchitecture-collecting-20-billion-metrics-a-day.html (дата звертання: 20.11.2024).

13. Prometheus vs Zabbix: відмінне та подібне цих систем моніторингую. URL: https://itedu.center/ua/blog/comparisons/prometheus_vs_zabbix/?srsltid=AfmBOoqapxCJC3O6io9A5herFuYZCrqMj52JlRODrWQLKan-HeLbQYiq (дата звертання: 22.11.2024).

14. Огляд системи мониторингу Zabbix. URL: http://www.flycat.info/2008/03/05/zabbix-monitoring-system (дата звертання: 22.11.2024).

15. Tysh Ie. Approach And Method Of Evaluation Of The General Reliability Indicator Of Computer Systems. International Scientific Journal Computer Systems And Information Technologies. Khmelnytskyi : Khmelnytskyi National University. №3. 2021. С.74-80.

16. Гладовський Н.О. Програмні рішення моніторингу систем та мереж // Інформаційні моделі, системи та технології: Праці XII наук.-техн. конф. (Тернопіль, ТНТУ ім. І. Пулюя, 18-19 грудня 2024 р.) с. 118.

17. Гладовський Н.О. Виявлення несправних параметрів у ході моніторингу мережі // Інформаційні моделі, системи та технології: Праці XII наук.-техн. конф. (Тернопіль, ТНТУ ім. І. Пулюя, 18-19 грудня 2024 р.) с. 119.

18. Palamar A., Palamar M., Osukhivska H. Real-time Health Monitoring Computer System Based on Internet of Medical Things. CEUR Workshop Proceedings, 3rd International Workshop on Information Technologies: Theoretical and Applied

Problems (ITTAP 2023), Ternopil, Ukraine, Opole, Poland, November 22–24, 2023. Vol. 3628. P. 106-115.

19. Gitomer, D. H., Steinberg, L. S., and Mislevy, R. J. (1995). Diagnostic assessment of a troubleshooting skill in an intelligent tutoring system. In Nichols, P. D., Chipman, S. F., and Brennan, R. L. (eds.), Cognitively Diagnostic Assessment, Lawrence Erlbaum Associates, Hillsdale, NJ.

20. Newell, A., and Simon, H. A. (1972). Human Problem Solving, PrinticeHall, Englewood Cliffs, NJ.

21. Jonassen, D. H., and Hernandez-Serrano, J. (2002). Case-based reasoning and instructional design: Using stories to support problem solving. Educ. Technol. Res. Dev. 50(2):pp. 65–77.

22. Cormode G., Muthukrishnan S., Yi K. Algorithms for Distributed Functional Monitoring// Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms. San Francisco, California. 2008. P. 1076-1085.

23. Wuhib F., Dam M., Stadler R., Clemm A. Decentralized computation of threshold crossing alerts// Proc. 16th IEEE/IFIP International Workshop on Distributed Systems. Barcelona, Spain. 2005. Vol. 3775. P. 220-232.

24. Wuhib F., Stadler R., Clemm C. Decentralized service-level monitoring using network threshold crossing alerts// IEEE Communications Magazine. 2006. Vol. 44. No 10. P. 70-76.

25. Stallings W. SNMP, SNMPv2, SNMPv3, RMON1 and 2. — 3rd edition. AdisonWesley. 1998. 640 p.

26. Steinder M., Sethi A. S. Probabilistic Fault Diagnosis in Communication Systems Through Incremental Hypothesis Updating// Computer Networks. July 2004. vol. 45. no. 4. pp. 537-562.

27. Appleby K., Goldszmidt G., Steinder M. Yemanja – A Layered Event Correlation Engine for Multi- domain Server Farms// Integrated Network Management Proceedings, 2001 IEEE/IFIP International Symposium on. 2001. pp. 329-344.

28. Rish I., Brodie M., Odintsova N., Ma S., Grabarnik G. Real-time Problem Determination in Distributed Systems using Active Probing// Network Operations and Management Symposium. NOMS 2004. IEEE/IFIP. April 2004. Vol. 1. pp. 133-146.

29. Guo J., Kar G., Kermani P. Approaches to Building Self Healing System 72 using Dependency Analysis// Network Operations and Management Symposium. NOMS 2004. IEEE/IFIP. April 2004. Vol. 1. pp. 119-132.

30. Tang Y., Al-Shaer E. S. Boutaba R. Active Integrated Fault Localization in Communication Networks// Integrated Network Management Proceedings. IM'2005. IEEE/IFIP International Symposium on. May 2005. pp. 543-556.

31. Автоматизація оновлення програмного забезпечення розподілених інфокомунікаційних систем. URL: http://www.tech.vernadskyjournals.in.ua/ journals/2024/1_2024/part_1/45.pdf (дата звертання: 28.11.2024).

32. Raviv Y., Intrator N. Bootstrapping with noise: An effective regularization technique. Connection Science 8 (3-4). Tel Aviv, 1996. P. 355-372.

33. Дубініна С. В. Байєсівські методи моделювання актуальних процесів та оцінювання ризиків страхових компаній: дис. канд. техн. наук: 05.13.23. Київ, 2017. 199 с.

34. Ensemble methods: bagging, boosting and stacking. URL: https://towardsdatascience.com/ensemble-methods-baggingboosting-and-stacking-c9214a10a205 (дата звернення:29.11.2024).

35. Skorenkyy Y., Kozak R., Zagorodna N., Kramar O., Baran, I. Use of augmented reality-enabled prototyping of cyber-physical systems for improving cybersecurity education. Journal of Physics: Conference Series. 2021. Vol. 1840, No. 1. DOI: http://dx.doi.org/10.1088/1742-6596/1840/1/012026

36. Guo Chen, Youjian Zhao, Dan Pei, and Dan Li. 2015. Rewiring 2 Links is Enough: Accelerating Failure Recovery in Production Data Center Networks. In Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on. IEEE, 569–578.

37. "Switch failure causes outages at hosting.com data center,". URL: http://www.datacenterdynamics.com/content-tracks/servers- storage/switch-

failurecauses-outages-at-hostingcom-data- center/32344.fullarticle. (дата звертання: 29.11.2024).

38. Лупенко С. А., Пасічник В. В., Тиш Є. В. Комп’ютерна логіка. Львів: Видавництво «Магнолія - 2006». 2015. 354 с

39. Liu Dapeng, Zhao Youjian, Xu Haowen, Sun Yongqian, Pei Dan, Luo Jiao, Jing Xiaowei, and Feng Mei. 2015. Opprentice: Towards Practical and Automatic Anomaly Detection through Machine Learning. In ACM IMC. Tokyo, Japan.

40. Заікіна Д., Глива В. Основи охорони праці та безпека життєдіяльності. 2019. URL: https://doi.org/10.31435/rsglobal/001 (дата звернення: 14.12.2024).

41. Безпека в надзвичайних ситуаціях. Методичний посібник для здобувачів освітнього ступеня «магістр» всіх спеціальностей денної та заочної (дистанційної) форм навчання / укл.: Стручок В. С. Тернопіль: ФОП Паляниця В. А., 2022. 156 с.

ДОДАТОК А

Тези конференції

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18-19 грудня 2024 року

ТЕРНОПІЛЬ
2024

УДК 004.94
Н. О. Гладовський
Тернопільський національний технічний університет імені Івана Пулюя, Україна

ВИЯВЛЕННЯ НЕСПРАВНИХ ПАРАМЕТРІВ У ХОДІ МОНІТОРИНГУ МЕРЕЖІ

N.M.Holovetskyi

DETECTION OF FAULTY PARAMETERS DURING NETWORK MONITORING

Розглянемо постановку задачі моніторингу комп'ютерної мережі. Загалом завдання моніторингу можна описати в такий спосіб: події, що відбуваються в комп'ютерних мережах та інформаційних системах, мають випадковий характер. У зв'язку з цим для їх вивчення найбільше підходять ймовірнісні математичні моделі теорії масового обслуговування, наприклад теорія марковських ланцюгів [1].

Для моделювання спочатку необхідно визначити вхідні дані. Розглянемо події, пов'язані з детектуванням аномалій у процесі діагностики та контролю досліджуваних системою параметрів обладнання в мережі, як стан аналізованої системи моніторингу мережі. Виходячи з викладеного вище, виведемо можливі стани в процесі моніторингу метрик (параметрів) мережного обладнання: S_0 – відсутність несправностей; S_1 - навантаження мережі; S_2 – зниження пропускної спроможності; S_3 - відсутність мережного порту; S_4 – фізична недоступність пристрою; S_5 - фрагментація пакета; S_6 - повна відмова системи.

Описаний ланцюг подій під час моніторингу мережі можна подати у вигляді графа станів, як показано на рис. 1.

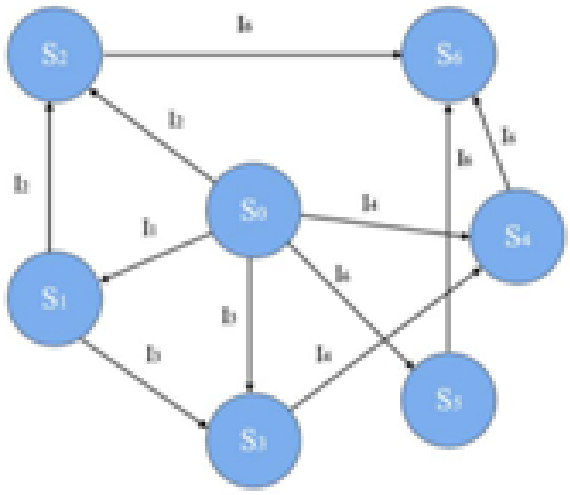

Рис. 1 - Граф виявлення несправних параметрів у ході моніторингу мережі

Імовірність виявлення несправного параметра в мережі визначається в даному випадку як ймовірність знаходження системи в стані S_i.

На графі кожна стрілка відповідає інтенсивності потоку подій, який переводить систему з одного стану в інший. Послідовність опитування визначається призначеним системним адміністратором пріоритетом. Чим вищий пріоритет у параметра мережного пристрою, тим вище інтенсивність його опитування. За інтенсивності переходів у разі приймається I_i - інтенсивність опитування i -го параметра у системі моніторингу.

Зважаючи на описані вище метрики роботи, система моніторингу виходить, що вона зі стану S_0 при виконанні опитування мережевих пристроїв при відмові будь-якого компонента переходить у стан виявлення несправностей S_i з інтенсивністю I_i. За графом можна скласти математичну модель процесу моніторингу мережі з урахуванням ймовірностей p_i як системи рівнянь. Її розв'язок дозволяє відстежити динаміку діагностування несправних параметрів у процесі мережного моніторингу через відстеження ймовірностей у певні проміжки часу.

1. Погоруй А. О., Чемерис О.А. Вступ до теорії випадкових процесів : навчальний посібник. Житомир : Вид-во ЖДУ ім. І. Франка, 2020. 70 с.

УДК 004.45
Н. О. Гладовський
Тернопільський національний технічний університет імені Івана Пулюя, Україна

ПРОГРАМНІ РІШЕННЯ МОНІТОРИНГУ СИСТЕМ ТА МЕРЕЖ

N.M.Holovetskyi

SYSTEM AND NETWORK MONITORING SOFTWARE SOLUTIONS

Системи моніторингу (СМ) дозволяють усувати незначні несправності, що становлять основну масу інцидентів, автоматично або за допомогою підключення віддаленим доступом до обладнання, що адмініструється.

Graphite. ПЗ застосовує тільки свою БД Whisper, а також є складним для налаштування. Проте є компенсація за рахунок високої продуктивності за великих даних обробки. Graphite створено моніторингу системних та бізнес-метрик, котрі реалізовані із використанням надбудов, у т.ч. і для прогнозування. Недолік продукту полягає у специфіці реалізації функцій під чітко визначені бізнес-процеси., відсутність автоматичного відтворення сервісів вслід за виявленням відмови. В цілому існують суб'єктивні та суперечливі думки про незручний інтерфейс додатків Graphite та вимушено високі витрати дуже великої кількості моніторингових серверів Graphite.

New Relic. Відома як програмний засіб відстеження та оцінювання застосунків, проте містить рішення також і для системних індексів із вбудованим сприянням фільтрації помилкових повідомлень та передбачення відмов. Біля 20 млн осіб застосовують розробки New Relic, котрі є доступними у хмарних середовищах, зокрема Amazone і Azure. Проте, безпекові умови компаній не дають змогу збереження даних, хоча б зашифрованих, на зовнішніх хмарних сховищах. Для вирішення цього New Relic дозволив під'єднувати свої модулі через JSON HTTP POST, і одержувати дані від серверів New Relic через API функцій та будувати власні екрани візуалізації даних.

Nagios. Проводить моніторинг за введеними вузлами та сервісами, та повідомляє мережевого адміністратора, якщо деякий сервіс чи вузол припиняють (або відновлюють) свою роботу. СМ має можливість створення карти мережі, із зазначенням детальної інформації про об'єкти. Особливості продукту: моніторинг стану робочих станцій (завантаження процесора, застосування простору жорсткого диска); інтуїтивно зрозуміла архітектура модулів розширень дає змогу застосовуючи будь-яку мову програмування, досить просто і легко створювати свої способи перевірки сервісів; є можливість інтеграції з іншими системами для покращення надійності і побудови розподіленої системи відстеження та оцінювання.

Prometheus. Є ще однією СМ різних сервісів та систем, при допомозі котрої сисадміни здатні збирати дані щодо метрик систем, налаштовувати додаткові функції для налаштування додаткових повідомлень (наприклад, повідомлення про те, коли виникають проблеми). Prometheus здобув популярність серед розробників, оскільки це проект з відкритим вихідним кодом, більшість компонентів якого написана мовою програмування Golang, а частина мовою Ruby. Виходячи з цього можна вивести, що у користувача даною системою буде лише один бінарний файл, який необхідно буде завантажити і встановити разом з компонентами системи Prometheus,

Zabbix. Універсальна СМ, що дозволяє спостерігати різні параметри на віддалених машинах. Параметри заносяться в БД, які при необхідності можна витягти, наприклад, для побудови графіків, налаштування оповіщення при переході значень параметрів через межу та ін. який знаходиться під моніторингом, опис проблеми, тривалість несправного стану та резолюція щодо усунення несправності. Оповіщення можуть здійснюватися як електронною поштою, так і через SMS.

ДОДАТОК Б

Фрагмент початкового коду скрипта loraDir.py

```
import simpy
import random
import npmpy as np
import ir.ath import sys
import matplotlib.pyplot as plt
import os



# turn on/off graphics
graphics = 0



# do the full collision check
full_collision = False
# experiments:
# 0: packet with longest airtime, aloha-style experiment
# 0: one with 3 frequencies,   1 with 1 frequency
# 2: with shortest packets, still aloha-style
# 3: with shortest possible packets, depending on distance



# this is an array with measured values for sensitivity
# see paper, Table 3
    SF7 = np.array ([ 7,- 126.5 ,- 124.125 ,- 120.75])
sf8 = np. array([8,-127.25,-126.75,-124.0]}
sf9 = np.array([9,-131.25,-128.25,-127.5])
sf10 = np.array([10, -132.75,-130,25,-128,75])
sfll = np. array ( 11. -134.5,-132.75,-120.75])
sf 12 = np.arrayl[12,-133.25,-132.25,-132.251)



#
# check for collisions at base station
# Note: called before a packet [ro rather node) is inserted in
the list def checkcol1ision (packet );
    col = 0 # flag needed since there might be several
collisions for packet
    processing = 0
    for i in range ( 0, len (packetsAtBS)) :
    if packetsAtBS[i].packet.processed == 1;
        processing = processing + 1
    if (processing > maxBSReceives) :
```