ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix»

на здобуття освітнього ступеня магістра
зі спеціальності 126 Інформаційні системи та технології
(код, найменування спеціальності)
освітньо-професійної програми Інформаційні системи та технології
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Володимир АРТЮШИН
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав:
здобувач вищої освіти Володимир АРТЮШИН
група ІСДМ-64

Керівник: Вікторія ЖЕБКА
науковий ступінь, вчене звання д.т.н., професор

Рецензент:
науковий ступінь, вчене звання _____
Ім'я, ПРІЗВИЩЕ

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інформаційних систем та технологій

Ступінь вищої освіти Магістр

Спеціальність Інформаційні системи та технології

Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ

_____Каміла СТОРЧАК
«_____»_____2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Артюшин Володимир Володимирович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix

керівник кваліфікаційної роботи Вікторія ЖЕБКА д.т.н., професор
(Ім'я, ПРІЗВИЩЕ науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційнихтехнологій від «__» __.2024р. №___

2. Строк подання кваліфікаційної роботи «____» грудня 2024р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, параметри моніторингу, розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз особливостей сучасних мобільних мереж та їх моніторингу.

Розробка моделі та постановка на моніторинг мережевих елементів по різним протоколам та баз даних.

Дослідження роботи створеної системи моніторингу та тестування результатів.

5. Перелік графічного матеріалу: *презентація*

1. Розробка моделі та постановка на моніторинг мережевих елементів по різним протоколам та баз даних.
2. Огляд інформаційної системи моніторингу
3. Тестування інформаційної системи моніторингу та огляд результатів

6. Дата видачі завдання «___» жовтня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	15.10 – 25.10.24	
2	Аналіз документації системи Zabbix	26.10 – 02.11.24	
3	Аналіз можливостей системи Zabbix	03.11 – 09.11.24	
5	Розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix	10.11 – 30.11.24	
6	Тестування інформаційної системи та виправлення помилок	01.12 – 11.12.24	
7	Оформлення роботи: вступ, висновки, реферат	12.12 – 20.12.24	
8	Розробка демонстраційних матеріалів	21.12 – 28.12.24	

Здобувач вищої освіти _____ Володимир АРТЮШИН
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Вікторія ЖЕБКА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 70 стор., 59 рис., 2 табл., 14 джерел.

Мета роботи – розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix.

Об'єкт дослідження – процес розробки інформаційної системи моніторингу мобільного оператора та налаштування її для моніторингу хостів системи.

Предмет дослідження – система моніторингу Zabbix, сервери з встановленою ОС Linux та мережевий комутатор.

Короткий зміст роботи: В результаті виконання кваліфікаційної роботи була розроблена інформаційна системи моніторингу мобільного оператора на базі системи Zabbix.
Для цього було проаналізовано основні проблеми сучасних мобільних мереж, типи обладнання, які використовуються мобільними операторами, сучасні системи моніторингу мобільного оператора та особливості системи Zabbix, як найбільш перспективної системи моніторингу для мереж мобільних операторів.
Щоб найбільш повно розкрити можливості системи Zabbix, була створена та встановлена на моніторинг модель системи білінгу міжоператорського трафіку, яка складається з трьох різних мережевих елементів з різними завданнями, протоколами та портами моніторингу: сервер з ОС Linux, сервер з встановленою БД Oracle та комутатор.

КЛЮЧОВІ СЛОВА: МОНІТОРИНГ, СИСТЕМА МОНІТОРИНГУ, ZABBIX, ZABIX АГЕНТ, МЕТРИКА, ТРИГЕР, МОБІЛЬНА МЕРЕЖА

ABSTRACT

Text part of the master's qualification work:
70 pages, 59 pictures, 2 table, 14 sources.

The purpose of the work - development of a mobile operator monitoring information system based on the Zabbix system.

Object of research – the process of developing a mobile operator monitoring information system and configuring it to monitor system hosts.

Subject of research – Zabbix monitoring system, servers with Linux OS installed, and a network switch.

Summary of the work: As a result of the qualification work, an information system for monitoring of mobile operator was developed based on the Zabbix system.
To this end, the main problems of modern mobile networks, types of equipment used by mobile operators, modern mobile operator monitoring systems, and features of the Zabbix system as the most promising monitoring system for mobile operator networks were analyzed.
To fully exploit the capabilities of the Zabbix system, a model of an inter-operator traffic billing system was created and installed for monitoring, which consists of three different network elements with different tasks, protocols, and monitoring ports: a server with the Linux OS, a server with an installed Oracle database, and a switch.

KEYWORDS: MONITORING, MONITORING SYSTEM, ZABBIX, ZABBIX AGENT, METRIC, TRIGGER, MOBILE NETWORK

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

3G	3-е покоління мобільних мереж
4G	4-е покоління мобільних мереж
5G	5-е покоління мобільних мереж
IoT	(Internet of Things, IoT) - Інтернет речей
LTE	(Long Term Evolution) - назва мобільного протоколу передачі даних
SIM	(Subscriber Identification Module) - Ідентифікаційний модуль абонента
ПЗ	(Software) - Програмне забезпечення, Програмні засоби
АЗ	(Hardware) - Апаратне забезпечення
OSS	(Operations Support System) - система експлуатаційної підтримки
NMS	(Network Management System) - система управління мережею
RAN	(Radio access network) - мережа радіодоступу
vRAN	(Virtual Radio Access Network) - віртуалізована мережа радіодоступу
PS Core	(Packet Switched Core) - мережа комутації пакетів
CS Core	(Circuit Switched Core) - мережа комутації каналів
IMS	(IP Multimedia Subsystem) - мультимедійна підсистема на основі протоколу IP
SDN	(Software-defined Networking) - програмно-конфігурована мережа
NFV	(Network Function Virtualization) - віртуалізація мережевих функцій
SNMP	(Simple Network Management Protocol) - простий протокол керування мережею зв'язку на основі архітектури TCP/IP
OID	(Object identifier) - Ідентифікатор об'єкта
LLD	(Low-level discovery) - Низькорівневе виявлення
TCP	(Transmission Control Protocol) - Протокол управління передачею
UDP	(User Datagram Protocol) - Протокол датаграм користувача

ВСТУП

Актуальність теми. Мобільна мережа оператора за останні два десятиліття перетворилася у велику та складну систему з різними типами обладнання та багаторівневою архітектурою. Навіть вихід з ладу одного мережевого елементу або каналу зв'язку між елементами може вплинути на працездатність сервісів або мережі в цілому. Саме тому побудова ефективної системи моніторингу є одним з пріоритетних завдань для операторів, що дає змогу швидко виявити аварійні ситуації та мінімізувати їх наслідки.

Мета і завдання дослідження є розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix.

Завдання дослідження:

1. Проаналізувати вхідні дані та вибрати метрики, які треба встановити на моніторинг;
2. Розробити інформаційну систему моніторингу мобільного оператора на базі системи Zabbix;
3. Дослідити роботу розробленої системи моніторингу та отримати результати тестування.

Об'єктом дослідження є процес розробки інформаційної системи моніторингу мобільного оператора та встановлення на моніторинг хостів системи.

Предметом дослідження є система моніторингу Zabbix версії 6.4, сервери з встановленою ОС Linux та мережевий комутатор.

Методи дослідження. У процесі роботи над проєктом були проаналізовані основні можливості системи моніторингу Zabbix, розроблена інформаційна система моніторингу мобільного оператора та продемонстровані отримані результати.

Апробація результатів та публікації. Результати досліджень було апробовано на II всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу» та II міжнародній науково-практичній

конференції «Сучасні аспекти діджиталізації та інформатизації в програмній та комп'ютерній інженерії».

Теоретична, методична та практична значущість отриманих результатів полягає в можливості застосування розробленої інформаційної системи моніторингу для удосконалення експлуатаційної діяльності та підвищення якості обслуговування мережевих елементів у середовищі мобільного оператора або провайдера. Розроблену систему можна використовувати як базовий шаблон для моніторингу основних показників.

1. ОСОБЛИВОСТІ СУЧАСНИХ МОБІЛЬНИХ МЕРЕЖ ТА ЇХ МОНІТОРИНГУ

1.1 Основні проблеми сучасних мобільних мереж

Основні проблеми сучасних мобільних мереж пов'язані з стрімким розвитком інформаційних технологій та зростанням попиту на послуги передачі даних.

Обсяг мобільного трафіку з роками зростає експоненціально. У середньому, кожні 1,5 – 2 роки трафік передачі даних у операторів подвоюється. Це залежить, також, від регіону та рівня проникнення цифрових послуг. У деяких регіонах із високою концентрацією користувачів мобільного інтернету, залежно від країни і рівня розвитку мережі, обсяг мобільного трафіку може подвоюватися навіть швидше – в середньому за 12 – 18 місяців.

Безпосередньо пов’язані з обсягом мобільного трафіку проблеми, це:

- перевантаження мережі;
- затримка і якість зв’язку;
- сумісність між поколіннями мереж;
- приватність та безпека.

Перевантаження мережі пов'язано зі збільшенням користувачів і пристроїв (зокрема, IoT), а також, збільшенням споживання трафіку у перерахунку на одного користувача (SIM-картку).

Затримка і якість зв’язку пов'язані з тим, що багато сучасних сервісів, як-от онлайн ігри, відеоконференції та відеотрансляції, вимагають високих швидкостей передачі даних та низької затримки. Однак, оператори мобільних мереж часто не можуть забезпечити відповідний рівень та якість сервісу.

Сумісність між поколіннями мереж потребує модернізації обладнання для забезпечення переходу абонента між різними стандартами, які на даний момент підтримуються мережею. Наприклад, абонент у процесі розмови або користуючись послугами передачі даних, може змінювати технологію 3G на 4G або 3G на 5G та навпаки.

Приватність та безпека також потребує особливої уваги у зв’язку з тим, що зі збільшенням підключених пристроїв зростають ризики вторгнення у конфіденційність даних користувачів або кібератак через мобільні мережі.

Крім вищезгаданих проблем, в умовах стрімко зростаючого трафіку, мобільний оператор повинен вчасно виконувати оновлення програмного (ПЗ) та апаратного забезпечення (АЗ), розширення ємності обладнання та ліцензійної ємності на самому обладнанні. Усі ці чинники вимагають від оператора постійних капіталовкладень у розвиток мережі.

Сучасна мережа мобільного оператора це велика та складна система з різними типами обладнання. Архітектура мережі оператора є багаторівневою системою, яка об'єднує різноманітні мережеві елементи з різними технологіями. Саме тому вихід з ладу окремого мережевого елементу може вплинути на працездатність мережі та окремих її сервісів, що у кінцевому випадку вплине на прибутки оператора та спричинити репутаційні ризики.

Для забезпечення безперебійної експлуатації мережі мобільного зв’язку операторами створюються окремі підрозділи у компанії – центри моніторингу, які цілодобово контролюють стан мережі, роботу обладнання, ключові показники якості мережі. Основними інструментами таких центрів виступають спеціалізовані системи моніторингу.

1.2 Сучасні системи моніторингу мобільного оператора

Сучасні системи моніторингу для мобільних операторів є високотехнологічними системами, що допомагають оператору підтримувати якість зв’язку, виявляти проблеми та оптимізувати роботу мережі.

Системи моніторингу мобільного оператора можуть включати у свій склад різні системи, наприклад:

- OSS (Operations Support System) - система експлуатаційної підтримки, яка забезпечує комплексне управління операціями мережі, включаючи роботу з різними видами мереж (мобільні, фіксовані, широкосмугові) та операціями, такими

як облік, налаштування, керування ресурсами, відновлення після збоїв, а також забезпечення якості обслуговування.

- NMS (Network Management System) - система управління мережею, яка відстежує продуктивність та стан окремих мережевих елементів різних підсистем. Вона використовується для моніторингу як апаратного, так і програмного забезпечення.
- CEM (Customer Experience Management) - система для управління досвідом користувача. Вона допомагає аналізувати якість обслуговування з боку абонентів і виявляти проблеми, що впливають на задоволеність клієнтів.
- Рішення на базі штучного інтелекту (AI та Machine Learning) - допомагають прогнозувати навантаження мережі, виявляти потенційні проблеми і відслідковувати аномалії в режимі реального часу.
- Big Data та аналітика - обробка великих даних, яка дозволяє операторам отримувати глибокі знання та напрацювання, аналізуючи трафік, поведінку та міграцію користувачів. Це дозволяє точно планувати ресурси та оптимізувати мережу.
- Рішення з кібербезпеки – зараз є дуже актуальними з огляду на високий рівень загроз у мережах мобільного зв’язку. Тому багато операторів інтегрують спеціалізовані системи моніторингу для відслідковування потенційних атак і загроз.

Для оперативного моніторингу стану мережі та обладнання центри моніторингу в основному використовують системи експлуатаційної підтримки (OSS) та системи управління мережею (NMS). Як вже було зазначено, OSS виконує ширший набір функцій, охоплюючи різні аспекти управління мережею та сервісами, а NMS зосереджується більше на моніторингу стану та продуктивності мережевих елементів і виявленні несправностей.

Слід зазначити, що мережеве обладнання у телекомі за АЗ та ПЗ поділяється на пропрієтарне та відкрите.

Під пропрієтарним обладнанням у телекомунікаціях розуміють обладнання та програмні рішення, які створені та контролюються певним виробником і

працюють на закритих (пропрієтарних) протоколах чи стандартах або архітектурі. Це означає, що таке обладнання часто не є сумісним з рішеннями інших виробників, а всі налаштування, оновлення та підтримка залежать від конкретного постачальника. Таке обладнання може контролюватися тільки спеціалізованими системами моніторингу від того ж постачальника обладнання.

Приміром може бути система Ericsson Network Manager (ENM) – сучасна система управління мережею радіодоступу (RAN), опорною (PS Core/CS Core/4G/5G) та мультимедійною (IMS) мережами оператора.

Компанія Nokia використовує платформу Nokia OSS (Operations Support Systems) для управління, моніторингу та оптимізації телекомунікаційних мереж. Основний продукт в рамках цієї системи – Nokia NetAct, який є комплексною платформою для управління мережевими ресурсами та надання послуг та виконує моніторинг та управління всіма елементами мережі, включаючи усі види 2G, 3G, 4G та 5G обладнання. Функціонально, NetAct – це система управління мережею (NMS).

Алалогічно, компанія Huawei використовує платформи OSS Huawei iManager U2000 та Huawei iManager U2020. Перша використовується для управління та моніторингу мобільними, фіксованими та IP-мережами. Тобто, елементами мережі для 2G, 3G, 4G, 5G, оптоволоконних та IP-мереж. Друга – для управління та моніторингу магістральними мережами та великими інфраструктурними транспортними мережами.

Так само, як Nokia, компанія ZTE використовує систему ZTE ZSmart OSS, яка є комплексною платформою для управління і моніторингу телекомунікаційних мереж та використовується для управління різними технологіями, включаючи мобільні мережі 2G, 3G, 4G та 5G, а також оптоволоконні та IP-мережі. Основний продукт в рамках цієї системи - ZTE NetNumen. Це система управління мережею (NMS) призначена для моніторингу, управління та оптимізації телекомунікаційних мереж. NetNumen підтримує різні типи мереж, включаючи фіксовані, мобільні та IP-мережі, що робить її гнучким і універсальним інструментом для операторів.

На противагу пропрієтарному обладнанню, мобільні оператори все частіше

починають використовувати універсальне обладнання, яке відповідає відкритим стандартам і може бути інтегроване з різними системами та від різних виробників. Використання такого обладнання сприяє гнучкості, зменшенню витрат та підвищенню сумісності мережевих рішень. Наприклад, таким обладнанням можуть бути:

1. Сервери загального призначення (COTS - Commercial Off-The-Shelf) – це стандартні сервери, які можна використовувати для різних завдань, зокрема для мережевих функцій (vRAN, SDN, NFV). Вони, зазвичай, підтримують різне програмне забезпечення і сумісні з більшістю віртуалізаційних рішень.

2. Базові станції на базі Open RAN – це базові станції, що використовують відкриті інтерфейси та стандарти, такі як O-RAN (Open Radio Access Network), що дозволяє інтегрувати обладнання різних виробників у єдину інфраструктуру.

3. Мережеві контролери SDN (Software-Defined Networking) – це програмні рішення, які дозволяють управляти мережею незалежно від виробника обладнання, використовуючи стандартизовані протоколи.

Процент використання універсального обладнання на мережі мобільного оператора може доходити до 100%.

На відміну від мереж мобільних операторів з пропрієтарним обладнанням, де для кожної підсистеми використовується окрема система моніторингу, для мереж, побудованих на універсальному або відкритому обладнанні, може бути застосована лише одна систем моніторингу для різних типів обладнання. Це значно знижує витрати оператора на обладнання та підвищує гнучкість мережі.

Системи OSS та NMS для управління універсальним обладнанням мобільних операторів виконують різні функції, такі як моніторинг мережі, управління послугами та ресурсами.

Прикладами систем моніторингу, які підходять для універсального або відкритого обладнання можуть бути наступні системи:

1. OpenNMS – система управління мережею з відкритим кодом, яка використовується для моніторингу і управління IT-інфраструктурою та мережевими ресурсами корпоративного рівня.

2. Zabbix – потужна система моніторингу з відкритим кодом, яка використовується для моніторингу статусів різноманітних сервісів комп'ютерної мережі, серверів та мережевого обладнання.

3. Prometheus – система моніторингу та збирання метрик з відкритим кодом, яка спеціалізується на зборі даних про продуктивність і стан інфраструктури в реальному часі. Вона і часто використовується для моніторингу контейнеризованих додатків (Kubernetes, Docker) і мікросервісів. Зазвичай його використовують разом із Grafana, що надає можливість створювати користувацькі дашборди з інтерактивними графіками і зручними для аналізу метриками.

4. NetXMS – відкрита система для моніторингу мережі та управління нею, яка підтримує різні типи обладнання.

5. ONAP (Open Network Automation Platform) – відкрита платформа автоматизації мережі, розроблена під егідою Linux Foundation. Використовується для автоматизації управління телекомунікаційною мережею та керування життєвим циклом мережевих сервісів, зокрема у віртуалізованих мережевих функцій (NFV) та програмно-конфігурованих мережах (SDN - Software-Defined Networking).

6. Nagios – відкрита система моніторингу, яка забезпечує контроль за станом мережевих пристроїв, серверів та додатків.

7. Cacti – інструмент для моніторингу мережі та візуалізації даних та моніторингу трафіку, що добре працює з відкритими стандартами.

8. OpenDaylight (ODL) – це платформа з відкритим кодом для управління програмно-конфігурованими мережами (SDN), яка забезпечує централізоване управління мережею через програмний інтерфейс. Вона широко використовується в телекомунікаціях, центрах обробки даних, хмарних середовищах і корпоративних мережах для оптимізації управління мережею.

Ці системи дозволяють оператору мобільного зв’язку працювати з обладнанням від різних виробників, швидко інтегрувати нові рішення та спрощувати управління мережею, завдяки чому оператор може легко масштабувати свою інфраструктуру та оптимізувати її витрати.

1.3 Особливості системи Zabbix

Як вже було відзнчено, Zabbix – це потужна система моніторингу з відкритим вихідним кодом, яка забезпечує повний контроль над інфраструктурою: мережевими пристроями, серверами, різними додатками та IT-ресурсами.

З основних особливостей системи Zabbix слід відмітити:

1. Підтримка різних методів збору даних. Система підтримує кілька методів збору даних, таких як:

- SNMP (Simple Network Management Protocol) для моніторингу мережевих пристроїв за допомогою пасивних методів збору даних, використовуючи трапери (trappers) так і активними методами, використовуючи полери (pollers). Підтримуються протоколи SNMP версій v1, v2 та v3;
- Agent-based моніторинг: встановлення агента Zabbix на пристрої для збору детальної інформації. Наприклад, навантаження на процесори, використання пам'яті, стан дисків, мережевий трафік, продуктивність додатків, утилізація швидкості інтерфейсів, тощо. Повнофункціональний та легко розширюваний агент розгортається на машинах, що спостерігаються. Можливо розгортання агенту як на Linux, так і на Windows серверах;
- Agentless моніторинг: моніторинг без агента через відкриті протоколи.
- Моніторинг JMX (Java Management Extensions) для моніторингу лічильників JMX у Java-додатках;
- Стан та доступність пристроїв IPMI (Intelligent Platform Management Interface).

2. Можливість ітеграції Zabbix з різними базами даних (БД).

3. Підтримка API розширює можливості інтеграції та обмін даних Zabbix та інших систем.

4. Гнучка система сповіщень і алертів Zabbix підтримує різні види сповіщень, включно з електронною поштою, SMS, месенджерами (Slack, Telegram тощо). Це

дозволяє швидко та комплексно реагувати на інциденти та налаштовувати повідомлення залежно від пріоритету та типу події.

5. Вебінтерфейс для адміністрування і налаштування та візуалізація даних, яка дозволяє створювати графіки, діаграми, карти, інформаційні панелі (dashboards) для візуалізації стану інфраструктури та по історії даних.

6. Zabbix добре масштабується і може використовуватися як у малих мережах, так і у великих розподілених системах. Система підтримує розподілену архітектуру з додатковими проксі-серверами, що дозволяє масштабувати моніторинг на велику кількість вузлів.

7. Управління ролями та доступом дозволяє налаштовувати права доступу для різних користувачів і груп. Це дає змогу забезпечувати безпеку системи та дозволяє надавати різні права доступу для різних користувачів та груп.

8. Гнучкі шаблони моніторингу дозволяють швидко налаштувати моніторинг різних типів пристроїв та сервісів. Можна використовувати стандартні шаблони або створювати власні для конкретних завдань. Шаблони можна завантажити з офіційного репозиторію Zabbix Share, який містить кілька сотень шаблонів для різних пристроїв і додатків.

9. Мережеве виявлення (Network Discovery) у Zabbix – це механізм автоматичного пошуку та додавання пристроїв і сервісів у мережі для моніторингу. Він дозволяє автоматизувати процес додавання нових пристроїв до системи Zabbix, що особливо корисно у великих мережах із динамічною інфраструктурою. Можна виконувати пошук за заданим діапазоном IP-адрес і шукати активні пристрої. Потім автоматично визначається типи пристроїв та, за потреби, автоматично прив’язуються відповідні шаблони до знайдених пристроїв. Також, можливо автоматично додати знайдені пристрої до відповідної групи хостів та налаштувати дії, які виконуватимуться при виявленні пристроїв, наприклад, створення тригерів, надсилання повідомлень або додавання певних тегів.

10. Виявлення низькорівневих елементів (Low-Level Discovery, LLD) – дозволяє автоматично виявляти ресурси та елементи всередині вже існуючих вузлів мережі. Це можуть бути, наприклад, файлові системи або мережеві інтерфейси.

LLD особливо корисне для моніторингу серверів зі складними та змінними файловими системами або інтерфейсами, які треба оперативно додавати на моніторинг або видаляти, якщо вони більше не виявляються впродовж заданого часу.

Таким чином, Zabbix надає мобільному оператору комплексне рішення для моніторингу всієї інфраструктури мережі та забезпечує оперативне виявлення та усунення проблем, що позитивно впливає на якість обслуговування клієнтів.

1.4 Постановка завдання дослідження

Основним завданням кваліфікаційної роботи є розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix.

Дистрибутив системи моніторингу Zabbix версії 6.4 встановлено на сервер під керуванням операційної системи Linux. Керування на налагодження моніторингу всіх мережевих елементів відбувається через WEB-інтерфейс.

Для прикладу розглядається система білінгу міжоператорського трафіку, яка складається з двох серверів на базі ОС Linux (interconnect.nss.cd та interconnect.dba.cd), на одному з яких встановлена база даних (БД) Oracle. Додатково, по протоколу SNMP моніториться мережевий комутатор interconnect.rtr.cd.

Для серверів на ОС Linux повинні моніторитися наступні показники (або метрики):

- доступність Zabbix агенту на хостах;
- метрики пов’язані з CPU (кількість процесорів, навантаження, утилізація);
- виявлення файлових систем та моніторинг вільного місця в кожній;
- моніторинг оперативної пам’яті (використана, загальна, процент утилізації);
- моніторинг розміру SWAP файлу (“файлу підкачки”);
- виявлення мережевих інтерфейсів, швидкості, пакетів з помилками та

стану кожного інтерфейсу;

Для серверу на якому встановлена база даних (БД) Oracle додатково необхідно моніторити утилізацію таблиць та доступність з'єднання з БД.

Комутатор interconnect.rtr.cd потрібно моніторити по протоколу SNMP, використовуючи полери або трапери, налаштовані на спеціальні ідентифікатор об'єкта (OID).

2. РОЗРОБКА МОДЕЛІ ТА ПОСТАНОВКА НА МОНІТОРИНГ МЕРЕЖЕВИХ ЕЛЕМЕНТІВ

2.1 Моніторинг Linux серверів

Для початку, треба створити вузел мережі у системі Zabbix. Для цього на самому вузлі мережі встановлюється Zabbix-агент, який сумісний з типом ОС (Windows або Linux) та співпадає з версією ПЗ Zabbix-серверу. У роботі будемо використовувати версію Zabbix-серверу 6.4. ПЗ для встановлення Zabbix-агенту можна знайти на офіційному сайті [11].

Після встановлення, на Zabbix-агенті треба виправити конфігураційний файл. В нашому випадку це файл /etc/zabbix/zabbix_agent2.conf. Параметром Hostmame=interconnect.nss.cd задаємо ім’я хоста, на який встановлено агент. Параметрами Server та ServerActive ми визначаємо адресу Zabbix серверу або Zabbix Proxy серверу.

Zabbix Proxy – це проміжний компонент системи моніторингу Zabbix, який використовується для збору даних із серверів, пристроїв чи інших об'єктів у віддалених мережах або для розподілу навантаження на основний Zabbix сервер. Proxy сервер є проміжним сервером між хостами що моніторяться та основним сервером Zabbix.

Для створення вузла мережі у системі Zabbix, треба у меню Web-інтерфейсу вибрати пункт “Збір даних”, потім “Вузли мережі” та натиснути кнопку “Створити вузел мережі”.

Після заповнення усіх необхідних полів меню, треба натиснути кнопку “Додати”, щоб створити новий вузол (Рис. 2.1). Особливу увагу слід приділити створенню інтерфейсів вузла. За замовчуванням Zabbix сервер з'єднується з Zabbix клієнтом (агентом) за допомогою протоколу TCP по порту 10050. Від Zabbix клієнту до Zabbix серверу дані надходять по іншому TCP порту - 10051.

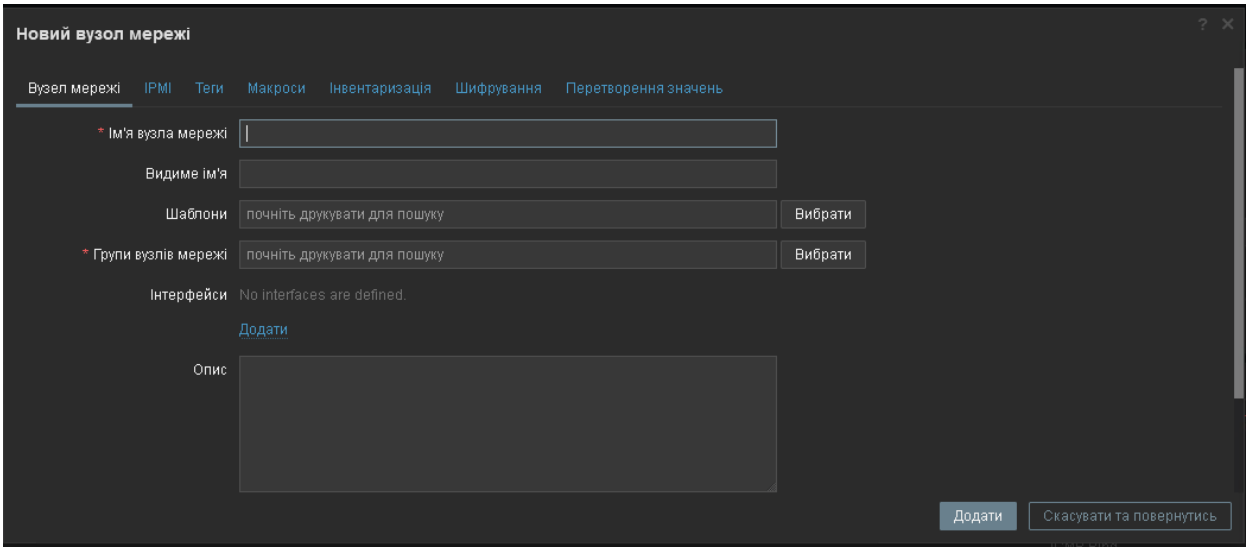


Рис. 2.1 Нове вікно створення вузла мережі у системі Zabbix

У полі “Шаблони” Рис приєднуємо до вузла необхідні шаблони (Рис. 2.2), які нам знадобляться для моніторингу необхідних метрик. Як вже було сказано, наразі доступні сотні шаблонів створених розробниками або спільнотою користувачів.

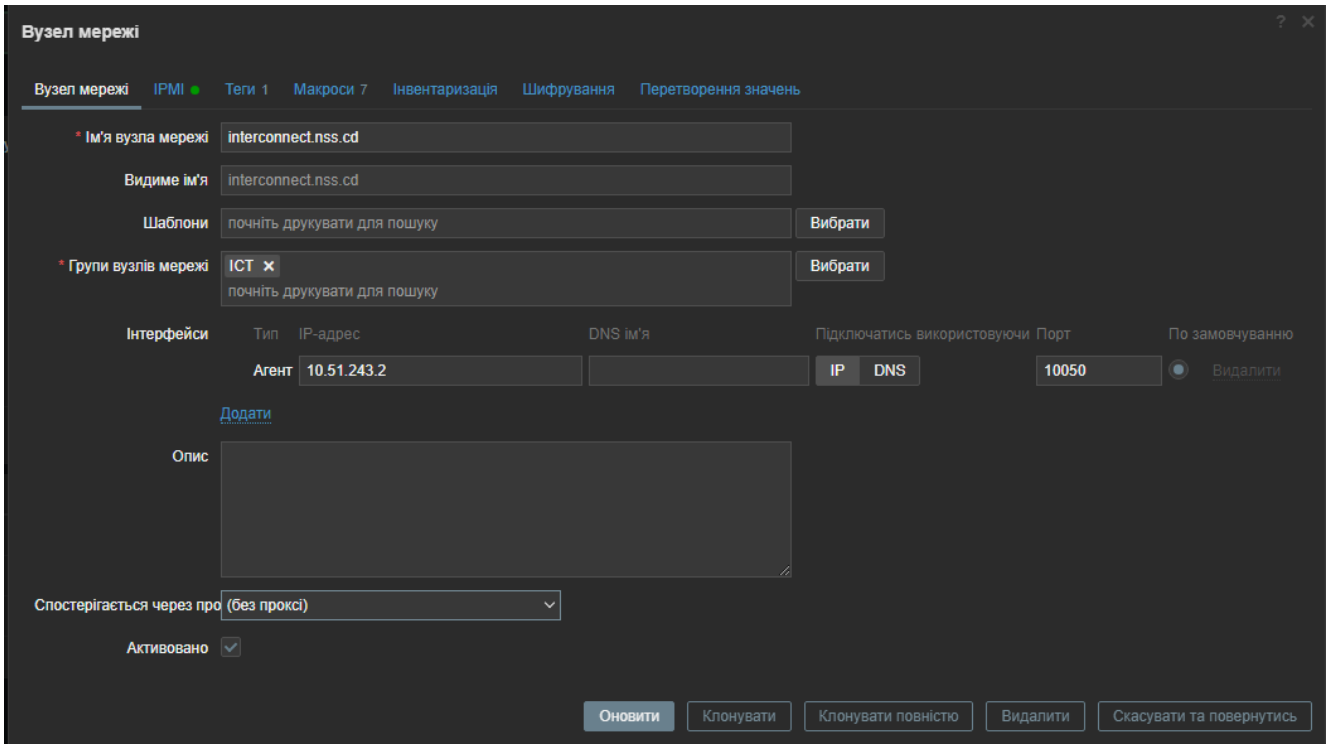


Рис. 2.2 Приклад створення вузла мережі у системі Zabbix

Основні метрики, які будуть встановлюватись на моніторинг наведені у Табл. 2.1

Таблиця 2.1

Основні метрики для моніторингу серверів з ОС Linux

Назва метрики	Опис технічного параметру	Одиниця виміру	Пороги та критичність
Memory utilization in %	Утилізація пам'яті	%	Попередження: > 90% за 5 хв. Критична: > 95% за 5 хв.
SWAP utilization in %	Утилізація файлу підкачки (SWAP) в %	%	Попередження: > 80% за 5 хв.
CPU total utilization in %	Утилізація процесору	%	Попередження: > 90% за 5 хв.
CPU load avarage in %	Середне навантаженя на процесор (5, 15 хв. інтервал)	число	Попередження: > 1.5
Space utilization in %	Використання простору в % (для кожного розділу файлової системи)	%	Попередження: util. > 85% Критична: util. > 95%
High inbound network utilization in %	Завантаження вхідного трафіку порівняно з максимальною пропускною здатністю мережевого інтерфейсу (для кожного мережевого інтерфейсу).	%	Попередження: > 90% за 10 хв.
High outbound network utilization in %	Завантаження вихідного трафіку порівняно з максимальною пропускною здатністю мережевого інтерфейсу (для кожного мережевого інтерфейсу).	%	Попередження: > 90% за 10 хв.
High error rate	Високий рівень помилок (для кожного мережевого інтерфейсу).	число	Попередження: > 5 за 5 хв.
Link down	Мережеве з’єднання не працює (для кожного мережевого інтерфейсу).	число	Попередження: = down

Для моніторингу метрик, наведених у Табл. 2.1 будемо використовувати елементи даних, які підтримується Zabbix агентом встановленим на хості, який знаходиться під моніторингом.

Метрику “Утилізація пам'яті” на Рис.2.3 отримуємо викликом елементу даних vm.memory.size[pavailable], де режим pavailable дозволяє отримати процент вільної пам’яті. Щоб отримати значення утилізациі пам’яті, тобто зайнятої пам’яті,

у елементі даних у вкладці “Попередня обробка” пишемо на JavaScript вираз, який віднімає процент вільної пам’яті від 100 (Рис. 2.4).

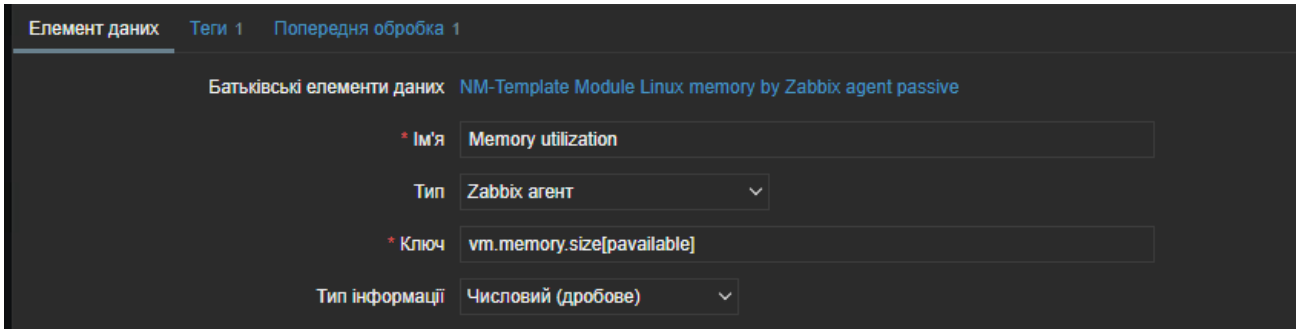


Рис. 2.3 Приклад створення елементу даних “Утилізація пам’яті”

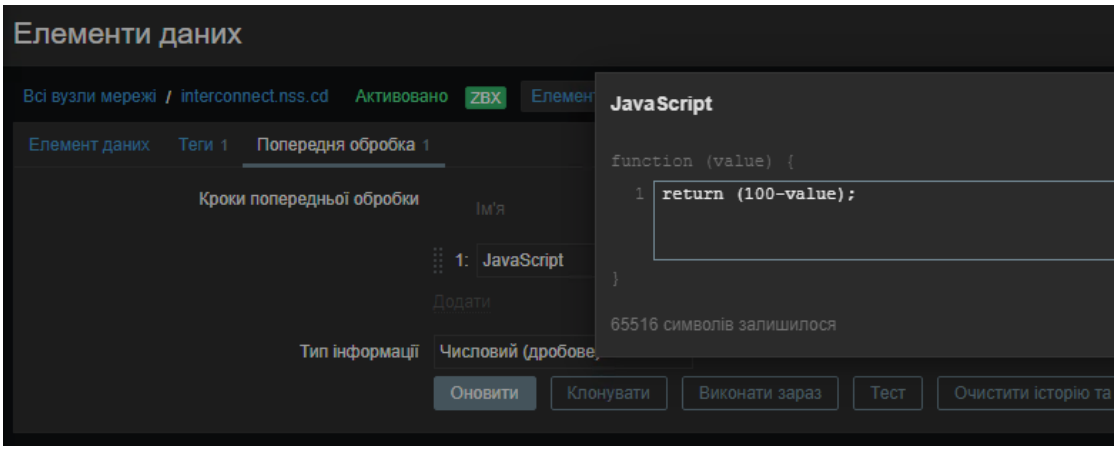


Рис. 2.4 Попередня обробка у елементі даних “Утилізація пам’яті”

Приклади тригерів для метрики “Утилізація пам'яті” з важливістю “Критичний” та “Попередження” наведені на Рис.2.5 та Рис. 2.6.

Для мінімізації хибних сповіщень, наприклад, коли виникають поодинокі випадки короткострокового перевищення критичних показників метрики, визначається мінімальний час тривалості аварії для спрацювання тригеру. Приймаємо, що критичним є, перевищення порогів 5-ти хвилин та більше (тобто, мінімальне значення метрики “Утилізація пам'яті” протягом 5 хвилин більше, ніж поріг спрацювання тригеру).

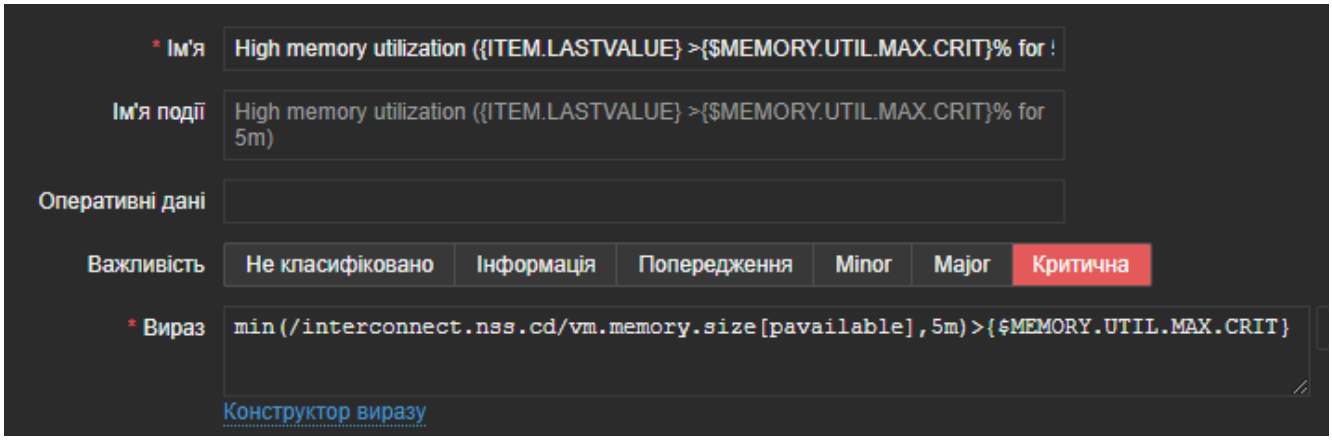


Рис. 2.5 Створення тригеру “Критичний” у елементі даних “Утилізація пам’яті”

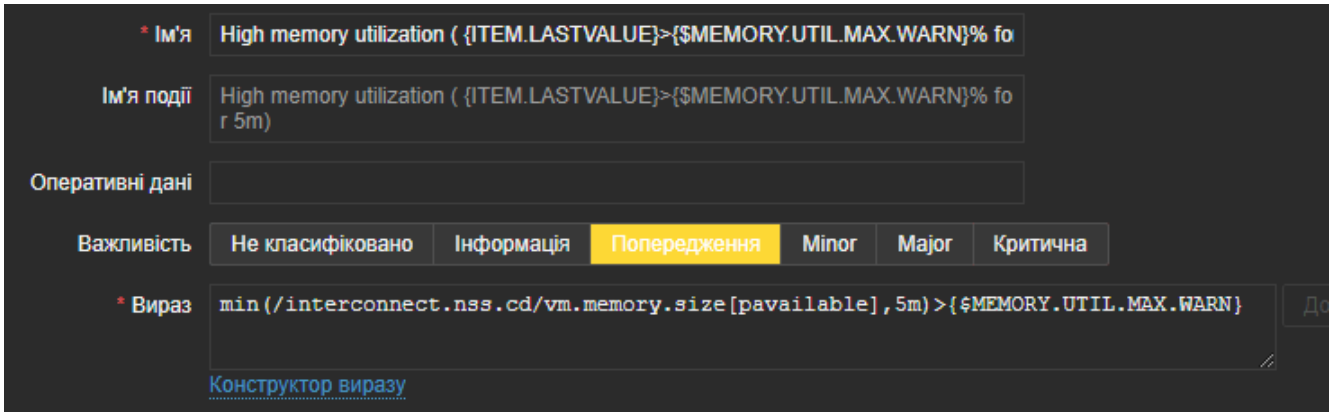


Рис. 2.6 Створення тригеру “Попередження” у елементі даних “Утилізація пам’яті”

Макроси у тригерах {$MEMORY.UTIL.MAX.CRIT.WARN} та {$MEMORY.UTIL.MAX.CRIT.CRIT} визначаються у макросах хосту та дорівнюють рівням критичності спрацювання трігерів – 90 та 95 процентів.

Аналогічним чином метрику “Утилізація файлу підкачки (SWAP) в %” з Табл 2.1 отримуємо викликом елементу даних system.swap.size[pfree] (Рис. 2.7). Повторюємо для метрики попередню обробку, як у елементі даних “Утилізація пам’яті” (Рис. 2.4), та створюємо такий саме тригер, але тільки з рівнем важливості “Попередження” (Рис. 2.6).

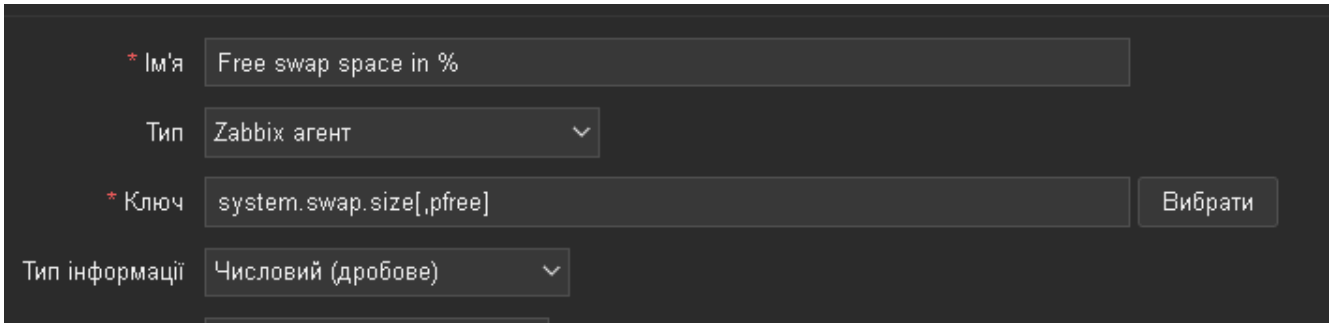


Рис. 2.7 Створення метрики “Утилізація файлу підкачки (SWAP) в %”

Метрика “Утилізація процесору” використовує елемент даних system.cpu.util[,idle] з попередньою обробкою віднімання даної метрики від 100. Це пов’язано з тим, що режим “idle” повертає процент простою процесора (CPU idle) у відсотках (Рис. 2.8).

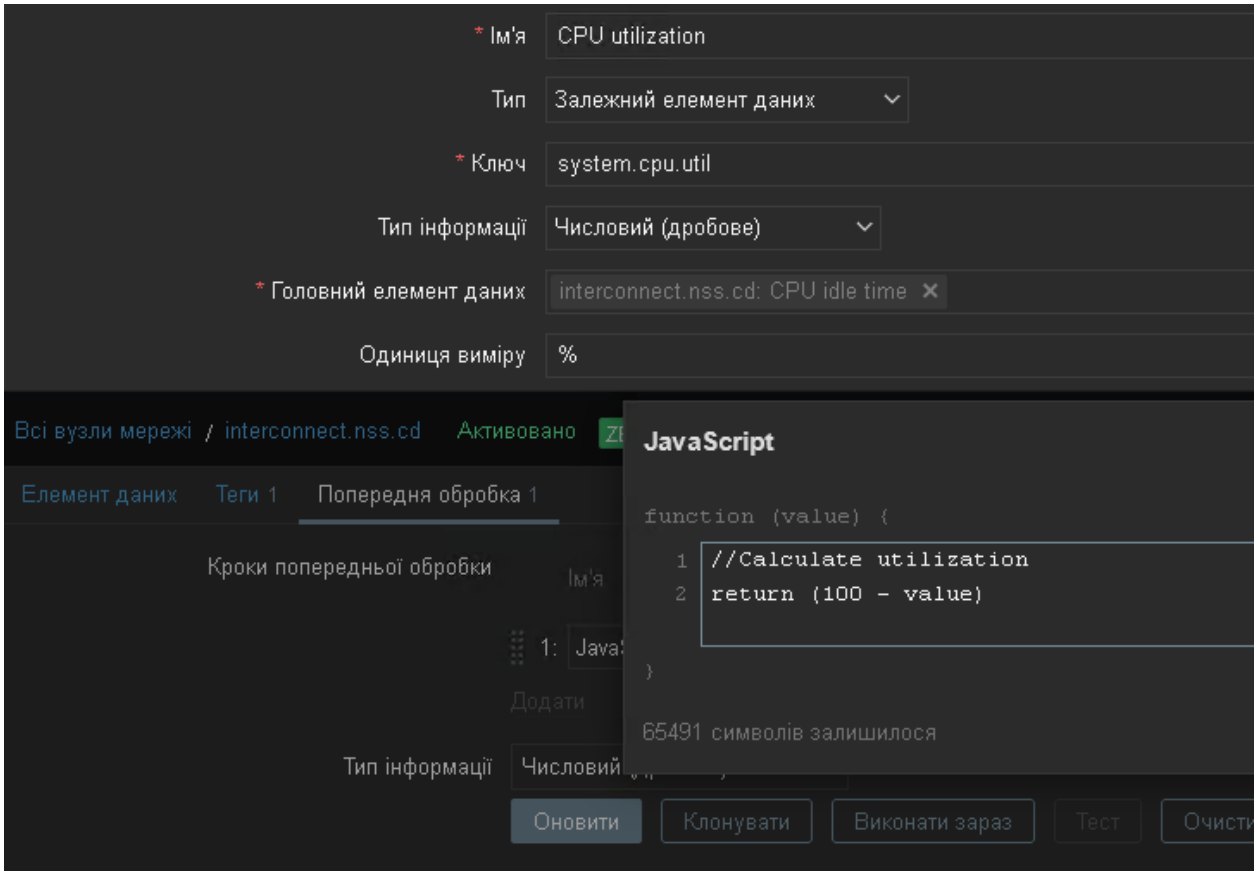


Рис. 2.8 Створення метрики “Утилізація процесору”

Середнє навантаженя на процесори (5, 15 хв. інтервал) отримуємо елементами даних system.cpu.load[all,avg5] (Рис. 2.9) та system.cpu.load[all,avg15]

(Рис. 2.10). Завантаження CPU вимірюється у вигляді коефіцієнта, що відображає кількість процесів, які чекають на виконання або працюють на процесорі.

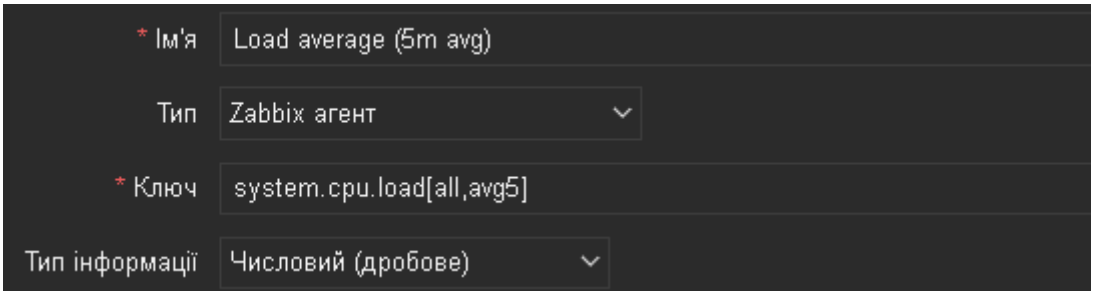


Рис. 2.9 Створення метрики середне навантаженя на процесори 5 хвилинний інтервал)

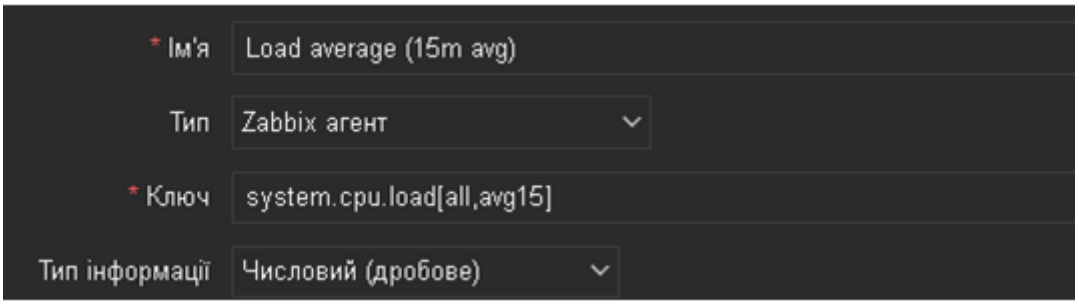


Рис. 2.10 Створення метрики середне навантаженя на процесори 15 хвилинний інтервал)

Розшифровка елементу даних:

system.cpu.load - це елемент, який відповідає за отримання даних про завантаження CPU;

all - визначає, що завантаження збирається для всіх ядер процесора. (можливий варіант - percpu, вказує на окремі ядра);

avg15 - середнє значення завантаження CPU за останні 15 хвилин;

avg5 - середнє значення завантаження CPU за останні 5 хвилин.

У тригері для середнього навантаження на процесор будемо враховувати, що ми отримали середнє навантаження для всіх CPU. Тому, для розрахунків треба це значення поділити на кількість процесорів у системі, яку ми отримуємо метрикою з ключом system.cpu.num. Приклад тригеру для середнього навантаження на процесор наведено на Рис. 2.11.

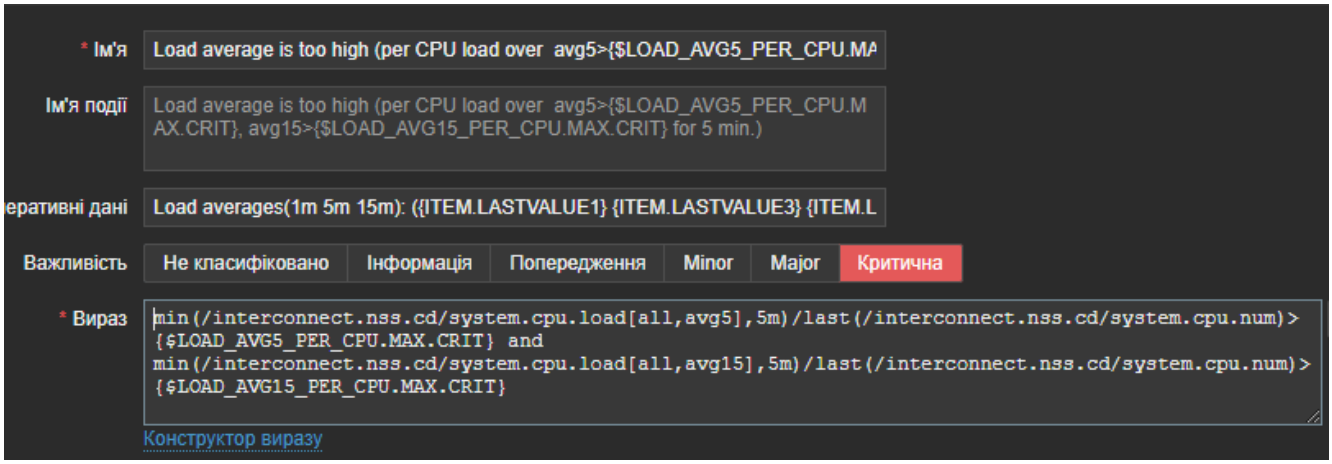


Рис. 2.11 Створення тригеру “Критичний”у елементі даних
“ Середне навантаженя на процесор (5, 15 хв. інтервал)”

Щоб отримати дані для метрики “Space utilization in %” створюємо правило виявлення (Рис. 2.12), яке за допомогою Zabbix агенту, встановленому на хості, отримує інформацію про файлові системи.

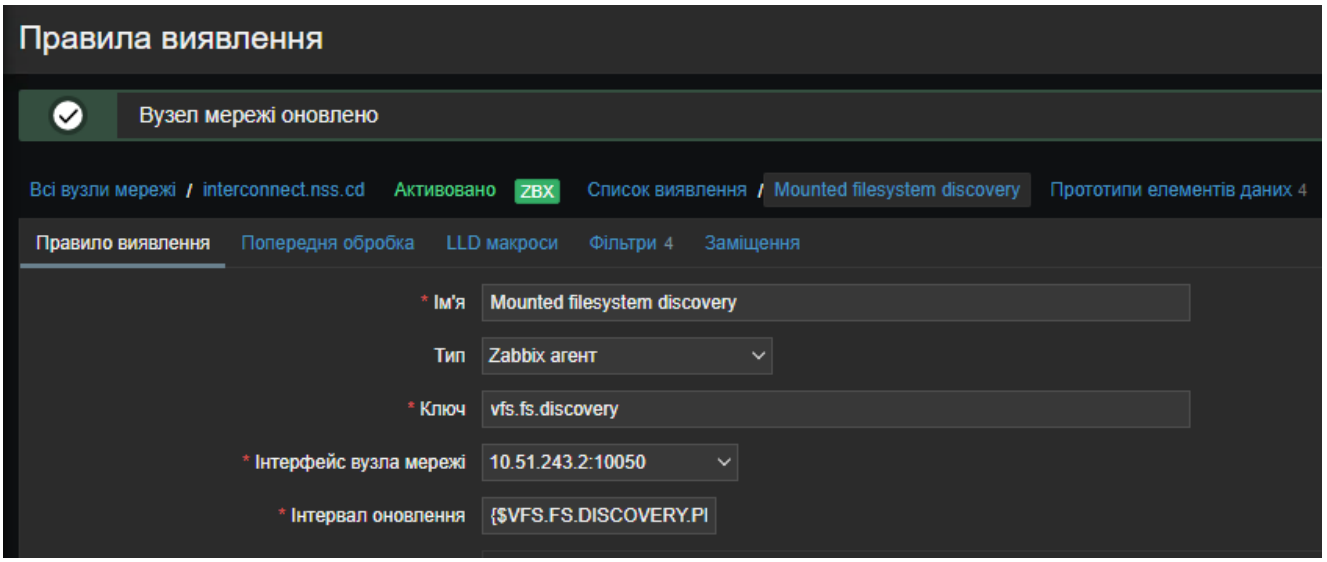


Рис. 2.12 Правило виявлення для отримання інформації з файлових систем

Нас цікавить назва файлової системи – {#FSNAME}, тому у прототипах елементів даних створюємо метрики відображені на Рис. 2.13, які відображають:

- Утилізацію файлової системи у % (ключ - pused);
- Сумарний розмір файлової системи у байтах (ключ - total);
- Використане місце файлової системи у байтах (ключ - used);

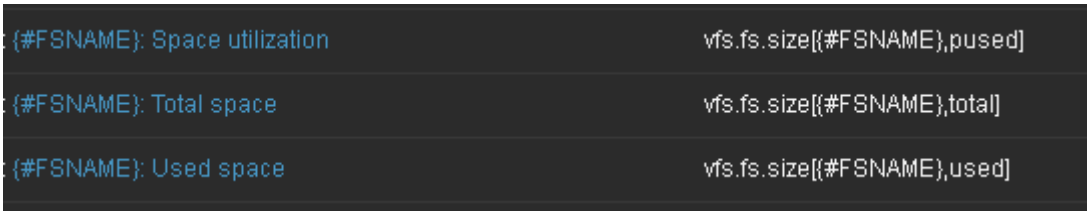


Рис. 2.13 Прототипи елементів даних для метрик з утилізації файлових систем

Назва елементів даних, які створюються вищезгаданим правилом виявлення на Рис. 2.14, містять унікальну назву яка складається з назв правила виявлення, файловой системи та метрики.

Рис. 2.14 Елементи даних для метрик з утилізації файлових систем

Тригери, створені правилом виявлення на Рис. 2.15, спрацьовують при перевищенні порогових значень заданих у макросах {$VFS.FS.PUSED.MAX.CRIT} для критичної важливості повідомлень та {$VFS.FS.PUSED.MAX.WARN} – для повідомлень з важливістю “Попередження”.

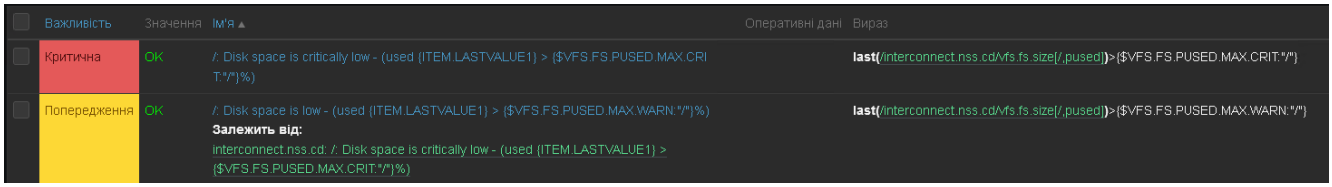


Рис. 2.15 Тригери для метрик з утилізації файлових систем

Останні чотири метрики у Табл 2.1 пов’язані з мережевими інтерфейсами серверу та можуть бути оброблені одним правилом виявлення. Щоб отримати

дані для метрик “High inbound network utilization in %”, “High outbound network utilization in %”, “High error rate” та “Link down” ми можемо створити правило виявлення або взяти готовий шаблон, який вже включає це правило виявлення (Рис. 2.16).

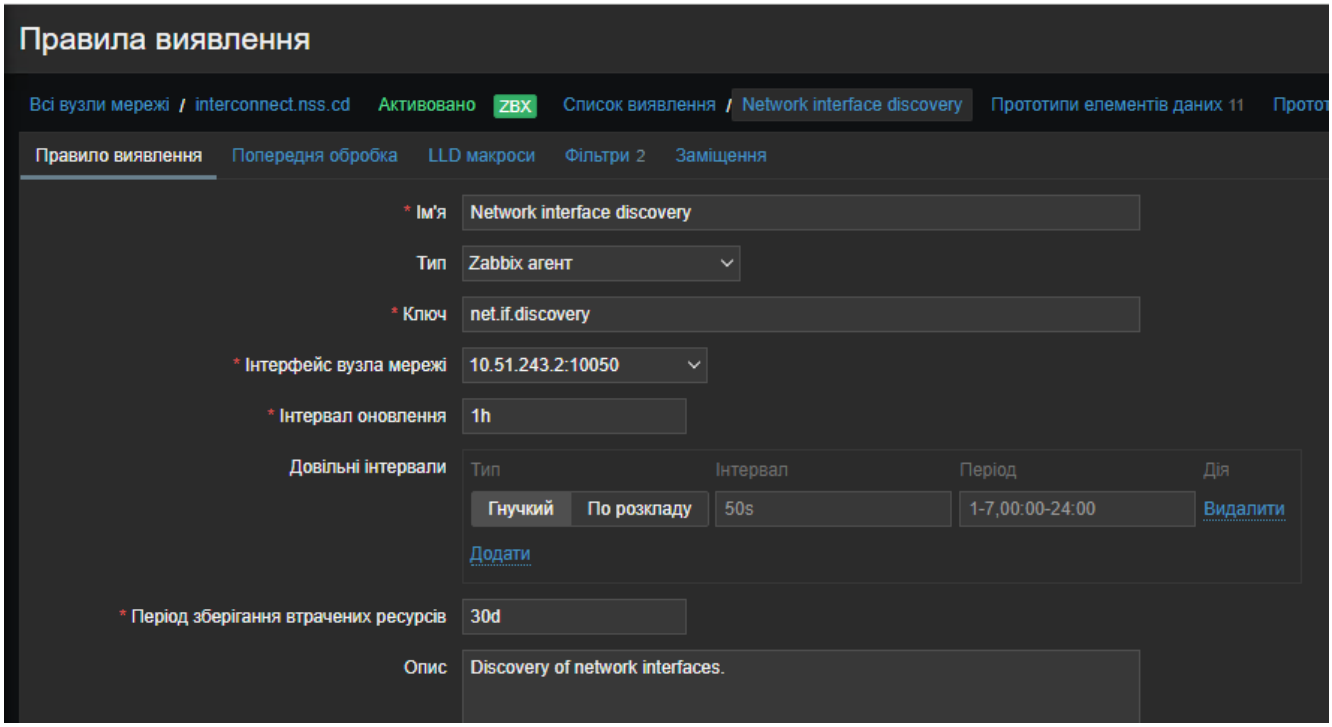


Рис. 2.16 Правило виявлення для отримання інформації з мережевих інтерфейсів

Як і з файловими системами, правило виявлення збирає всю інформацію про інтерфейси та привласнює кожному унікальний ключ з макросу {#IFNAME} у який записується назва інтерфейсу. Прототипи елементів даних створюються як показано на Рис. 2.17, а елементи даних, наприклад для інтерфейсу ens192, створюються у системі як показано на Рис.2.18.

Interface {#IFNAME}: Bits received per second	net.if.in["{#IFNAME}"]
Interface {#IFNAME}: Bits sent per second	net.if.out["{#IFNAME}"]
Interface {#IFNAME}: Inbound network utilization	net.if.in.util["{#IFNAME}"]
Interface {#IFNAME}: Inbound packets discarded	net.if.in["{#IFNAME}",dropped]
Interface {#IFNAME}: Inbound packets with errors	net.if.in["{#IFNAME}",errors]
Interface {#IFNAME}: Interface speed, bps	vfs.file.contents["/sys/class/net/{#IFNAME}/speed"]
Interface {#IFNAME}: Interface type	vfs.file.contents["/sys/class/net/{#IFNAME}/type"]
Interface {#IFNAME}: Operational status	vfs.file.contents["/sys/class/net/{#IFNAME}/operstate"]
Interface {#IFNAME}: Outbound network utilization	net.if.out.util["{#IFNAME}"]
Interface {#IFNAME}: Outbound packets discarded	net.if.out["{#IFNAME}",dropped]
Interface {#IFNAME}: Outbound packets with errors	net.if.out["{#IFNAME}",errors]

Рис. 2.17 Прототипи елементів даних для метрик, пов’язаних з мережевими інтерфейсами.

	Ім'я ▾	Тригери	Ключ
•••	Network interface discovery: Interface ens192: Outbound packets with errors	Тригери 1	net.if.out["ens192",errors]
•••	Network interface discovery: Interface ens192: Outbound packets discarded		net.if.out["ens192",dropped]
•••	Network interface discovery: Interface ens192: Outbound network utilization	Тригери 1	net.if.out.util["ens192"]
•••	Network interface discovery: Interface ens192: Operational status	Тригери 1	vfs.file.contents["/sys/class/net/ens192/operstate"]
•••	Network interface discovery: Interface ens192: Interface type		vfs.file.contents["/sys/class/net/ens192/type"]
•••	Network interface discovery: Interface ens192: Interface speed, bps		vfs.file.contents["/sys/class/net/ens192/speed"]
•••	Network interface discovery: Interface ens192: Inbound packets with errors	Тригери 1	net.if.in["ens192",errors]
•••	Network interface discovery: Interface ens192: Inbound packets discarded		net.if.in["ens192",dropped]
•••	Network interface discovery: Interface ens192: Inbound network utilization	Тригери 1	net.if.in.util["ens192"]
•••	Network interface discovery: Interface ens192: Bits sent per second		net.if.out["ens192"]
•••	Network interface discovery: Interface ens192: Bits received per second		net.if.in["ens192"]

Рис. 2.18 Приклад елементів даних для інтерфейсу “ens192”

Метрика “Interface {#IFNAME}: Inbound network utilization” (Рис. 2.19) вираховується автоматично по формулі:

(last(//net.if.in["{#IFNAME}"]) /

last(//vfs.file.contents["/sys/class/net/{#IFNAME}/speed"])) * 100

де last(//net.if.in["{#IFNAME}"] - елемент, який збирає останнє значення вхідного трафіку (в байтах за секунду) через мережевий інтерфейс {#IFNAME};

last(//vfs.file.contents["/sys/class/net/{#IFNAME}/speed"]) - елемент, який збирає швидкість мережевого інтерфейсу {#IFNAME}, яка зберігається у файлі /sys/class/net/{#IFNAME}/speed

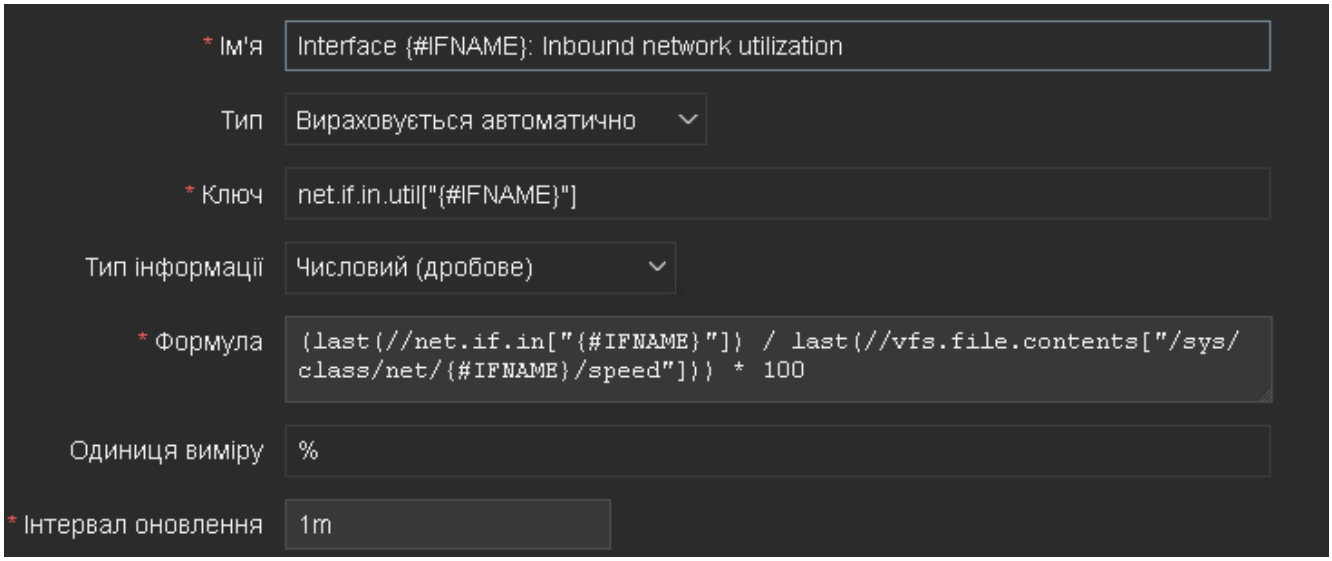


Рис. 2.19 Метрика “Interface {#IFNAME}: Inbound network utilization”

Подібним чином, метрика “Interface {#IFNAME}: Outbound network utilization” вираховується по формулі на Рис. 2.20.

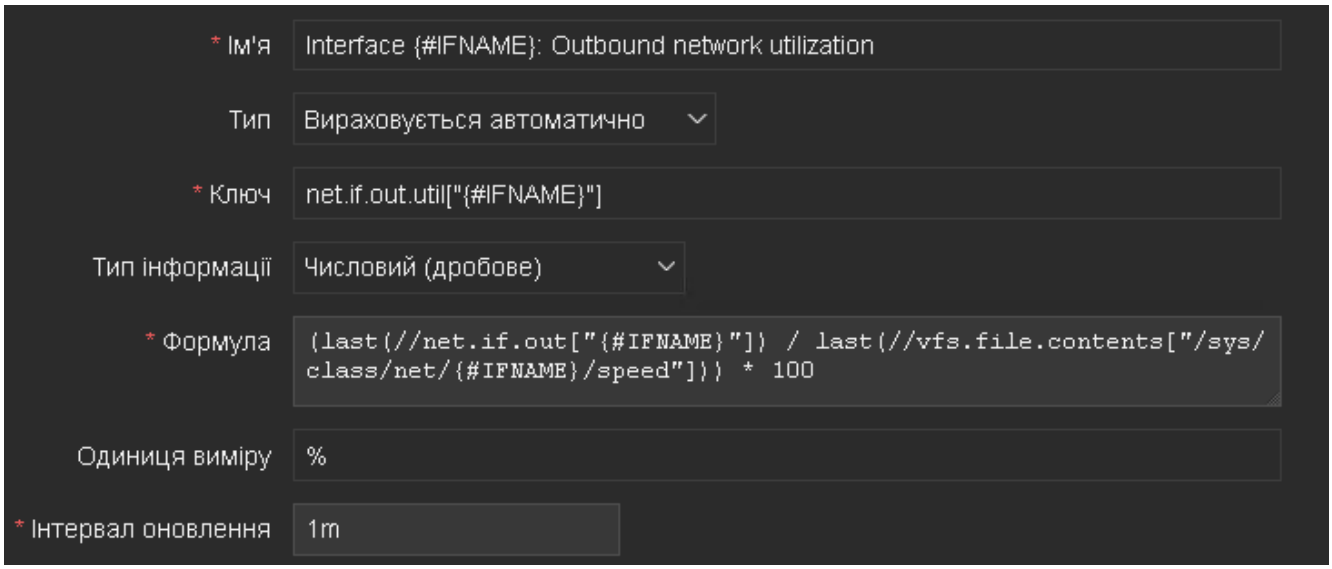


Рис. 2.20 Метрика “Interface {#IFNAME}: Outbound network utilization”

У формулі перша частина містить елемент даних last(//net.if.out["{#IFNAME}"],

який збирає останнє значення вхідного трафіку (в байтах за секунду) через мережевий інтерфейс {#IFNAME}, а друга частина формули аналогічна попередній метриці.

Створюємо тригери у прототипах правил виявлення мережевих інтерфейсів і, після виявлення, вони з'являються у елементах даних вже з назвами інтерфейсу замість макросу {#IFNAME}. На Рис. 2.21 вираз тригеру означає, що він спрацює при перевищенні порогового значення, заданого у макросу {$NET.UTIL.MAX} для повідомлень з важливістю "Попередження", через 10 хвилин після початку проблеми.

Важливість	Значення	Ім'я ▲	Вираз
Попередження	OK	Network interface discovery: Interface ens192: High inbound network utilization (>{$NET.UTIL.MAX}% for 10m)	min(/interconnect.nss.cd/net.if.in.util["ens192"],10m)>{$NET.UTIL.MAX}
Попередження	OK	Network interface discovery: Interface ens192: High outbound network utilization (>{$NET.UTIL.MAX}% for 10m)	min(/interconnect.nss.cd/net.if.out.util["ens192"],10m)>{$NET.UTIL.MAX}

Рис. 2.21 Приклад тригерів для метрик з утилізації мережевих інтерфейсів

Прототипи метрики "High error rate" (Рис 2.17) складаються з двох метрик для вхідних пакетів з помилками на інтерфейсі "Interface {#IFNAME}: Inbound packets with errors" з ключом net.if.in["{#IFNAME}",errors] та вихідних пакетів з помилками на інтерфейсі "Interface {#IFNAME}: Outbound packets with errors" з ключом net.if.out["{#IFNAME}",errors]. Прототипи тригерів для метрики "High error rate" (Рис 2.22) складаються з двох частин. Це вираз для проблеми, який спрацьовує, якщо кількість вхідних або вихідних помилок на інтерфейсі впродовж 5 хвилин буде перевищувати значення задане у макросі {$IF.ERRORS.WARN}, яке у нашому випадку дорівнює кількості 5 помилок.

Друга частина тригеру містить вираз для відновлення, яких закриває проблему. Він спрацьовує, якщо кількість вхідних або вихідних помилок на інтерфейсі впродовж 5 хвилин буде меньше на 20% ніж значення задане у макросі {$IF.ERRORS.WARN}. Тобто, не перевищувати 4 помилки.

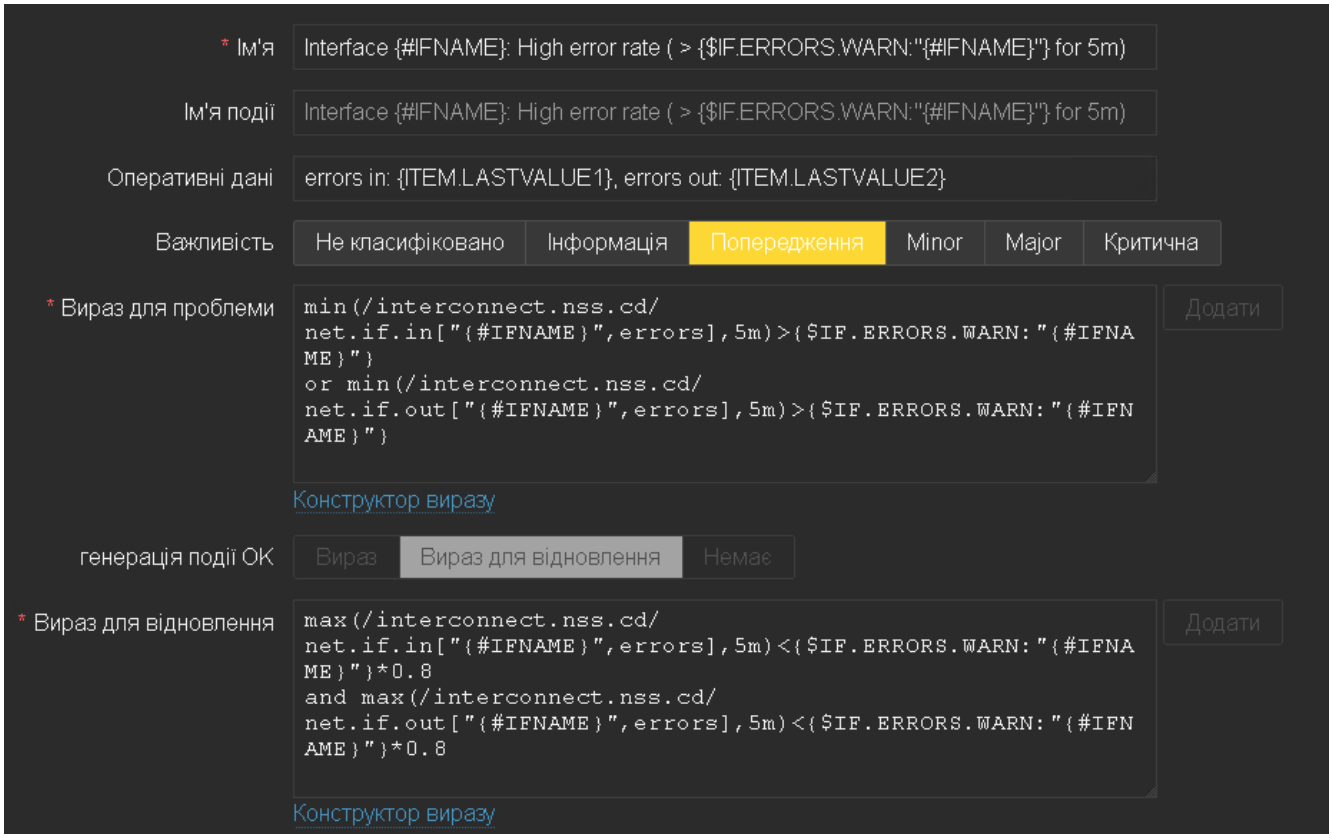


Рис. 2.22 Приклад тригерів для метрик з виявлення помилок на мережевих інтерфейсах

Для створення тригеру останньої метрики з Табл 2.1 “Мережеве з’єднання не працює” (або “Link Down”), треба використати прототип даних “Operational status” (Рис. 2.17). Приклад прототипу тригерів для метрик з виявлення непрацюючого мережевого з’єднання наведено на Рис. 2.23. Вираз “{$IFCONTROL:"{#IFNAME}"}=1” показує, що моніторинг або контроль увімкнено для конкретного інтерфейсу. Далі перевіряється стан мережевого інтерфейсу через файли ОС Linux /sys/class/net/... Якщо стан мережевого інтерфейсу дорівнює 2 (тобто “Link Down”) та є зміна метрики між останнім та передостаннім станом, то тригер увімкнеться.

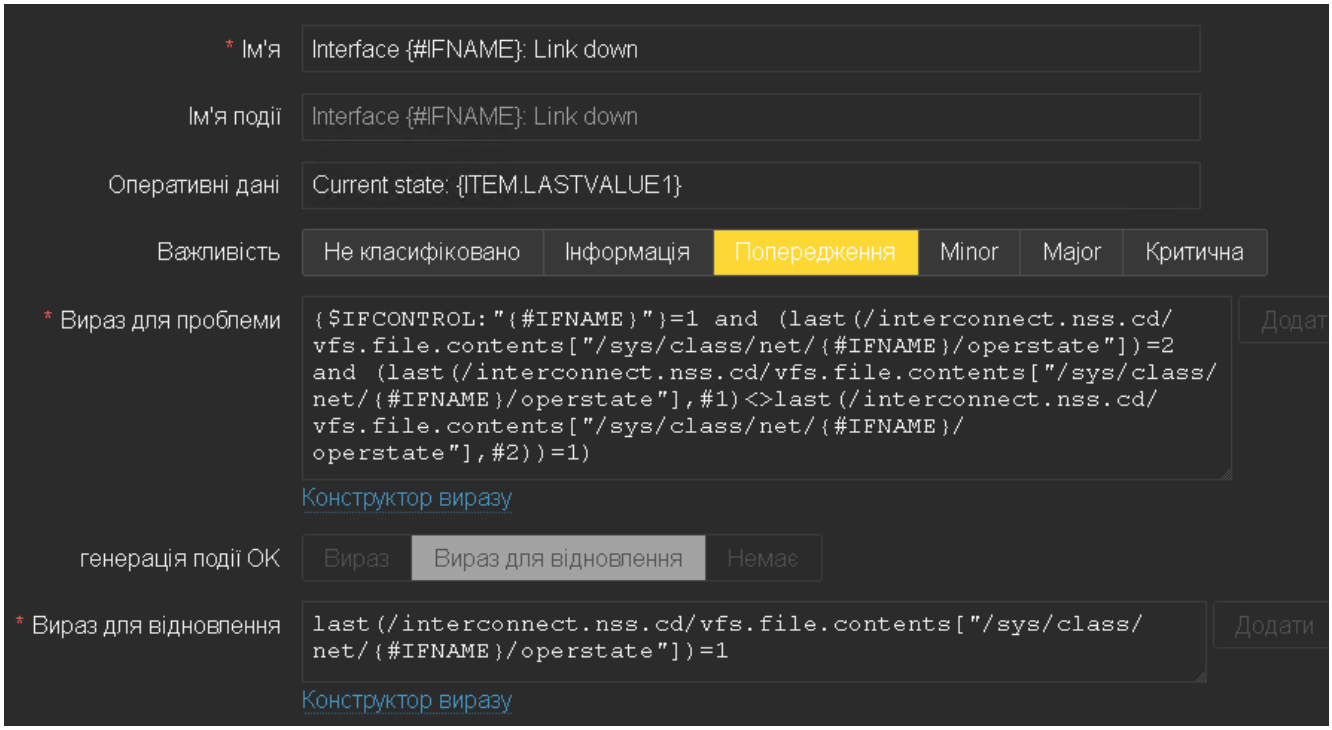


Рис. 2.23 Приклад прототипу тригерів для метрик з виявлення непрацюючого мережевого з’єднання

Тригер вимкнеться (Рис 2.23), якщо стан мережевого інтерфейсу буде дорівнювати 1 (тобто “Link Up”).

Таким чином ми встановили на моніторинг основні метрики, які були наведені у Табл. 2.1.

2.2 Моніторинг баз даних Oracle

Для серверу interconnect.dba.cd, на якому встановлена база даних (БД) Oracle. додатково треба організувати моніторинг доступності з'єднання з БД та утилізації таблиць.

Для моніторингу доступності з'єднання з БД, створюємо у системі Zabbix елемент даних “SQL connection status” типу “Зовнішня перевірка” (Рис. 2.24). Ключ – це shell скрипт, який буде виконуватися у спеціальній директорії з файлами …\externalscripts\ та додаткові параметри підключення до бази даних.

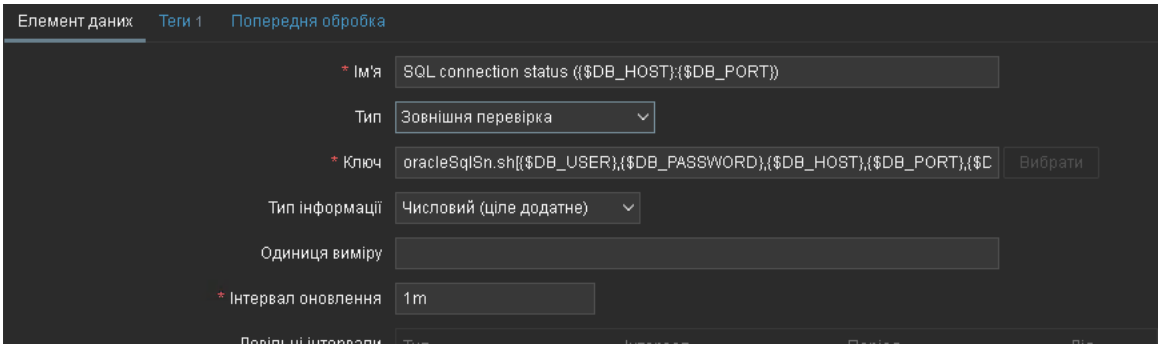


Рис. 2.24 Створення елементу даних “SQL connection status”

```
#!/usr/bin/bash

# Database connection parameters.
db_user=$1
db_password=$2
db_host=$3
db_port=$4
db_service_name=$5
db_sql=$6

# Connect to database using 'sqlplus' and select data.
sqlplus -S << EOF
${db_user}/${db_password}@${db_host}:${db_port}/${db_service_name}
$db_sql
quit;
EOF
```

Рис. 2.25 Зміст зовнішнього скрипту oracleSqlSn.sh

Скрипт на Рис. 2.25 запускає утиліту sqlplus, яка використовується для роботи з базами даних Oracle. Конструкція між символами << EOF та EOF (називається heredoc) дозволяє передати багаторядковий текст у вигляді вводу до команди. Змінні db_user (ім'я користувача бази даних), db_password (пароль користувача), db_host (хост, де розташована база даних), db_port (порт для підключення), db_service_name (ім'я сервісу бази даних), db_sql (SQL-запит, який потрібно виконати) передаються скрипту викликаючою програмою. У нашому випадку, елемент даних Zabbix на Рис. 2.24 кожну хвилину виконує виклик скрипту oracleSqlSn.sh з параметрами (ключом): oracleSqlSn.sh[{$DB_USER},{$DB_PASSWORD},{$DB_HOST},{$DB_PORT},{$DB_SERVICE_NAME},select count(*) from dual;]. Макроси у дужках {} прописані як макроси вузла мережі. Приклад макросів вузла наведено на Рис. 2.26.

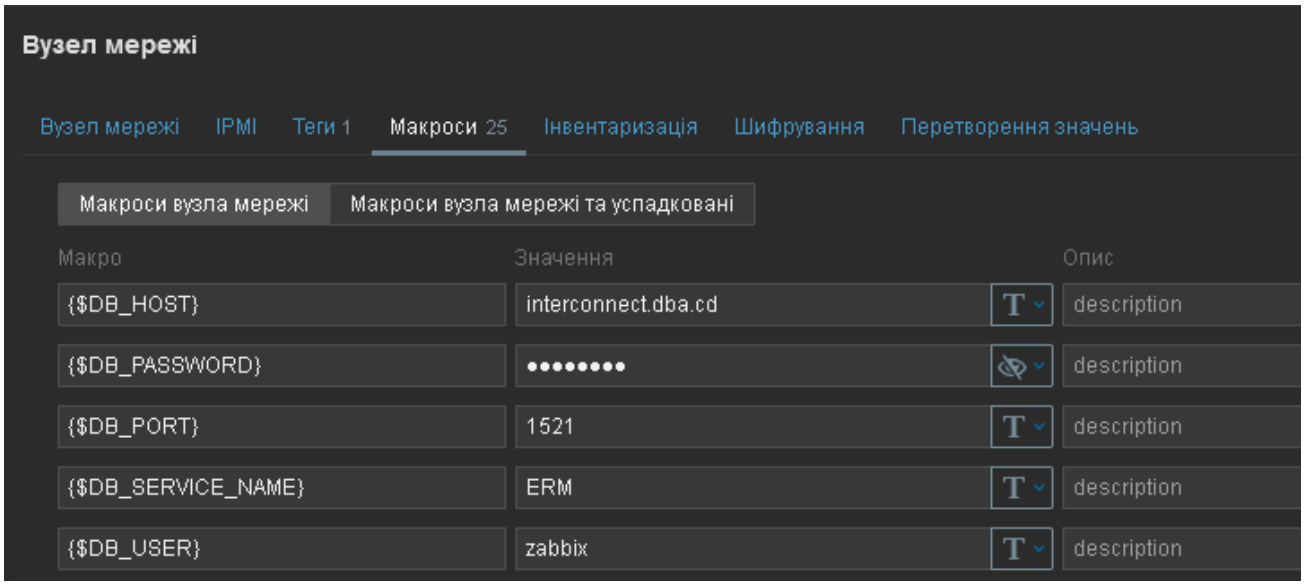


Рис. 2.26 Приклад макросів вузла мережі

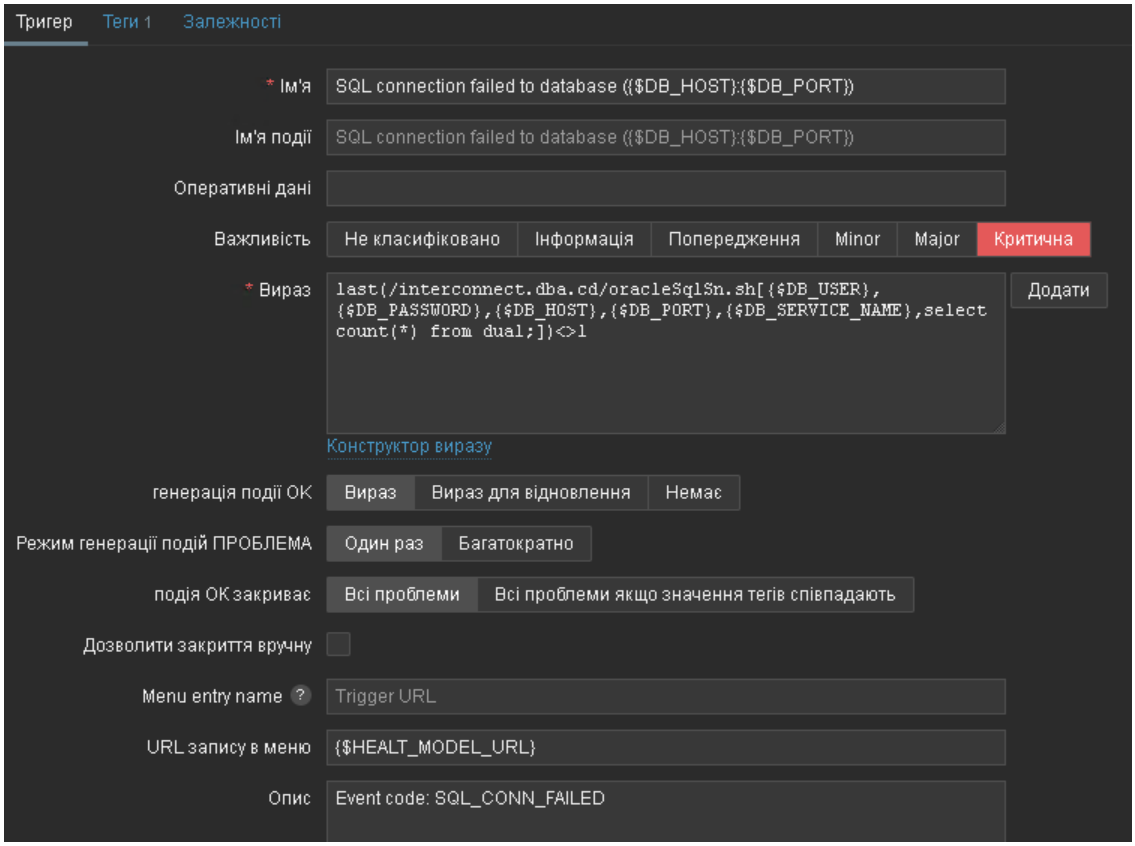


Рис. 2.27 Створення тригеру для елементу даних “SQL connection status”

SQL команда запиті “select count(*) from dual;” перевіряє кількість записів у спеціальній системній таблиці DUAL, яка містить лише один рядок, і створена для

виконання запитів, що не залежать від реальних таблиць. Таким чином, ми будемо отримувати значення 1, якщо є з'єднання з базою даних та 0 - якщо немає з'єднання.

Наступним кроком створюємо тригер зображений на Рис 2.27, який буде мати важливість "Критична" та спрацьовувати, коли значення метрики не буде дорівнювати числу 1.

Процес моніторингу утилізації БД Oracle будемо організовувати за допомогою низькорівневого виявлення (LLD), яке дає можливість автоматично створювати елементи даних, тригери та графіки навіть не знаючи наперед їх кількість та назви [12]. Крім того, якщо елемент даних більше не виявляється (наприклад, у БД була вилучена таблиця), ми можемо налаштувати автоматичне видалення непотрібних ресурсів.

Для початку створимо правило виявлення (Рис. 2.28) та прототип елементів даних (Рис. 2.29). У правилі виявлення відмічаємо період зберігання втрачених ресурсів – 1 годину. Вказуємо типи елементів даних "Zabbix трапер". Це означає, що елементи даних будуть приймати вхідні дані визначеного формату замість запиту цих даних.

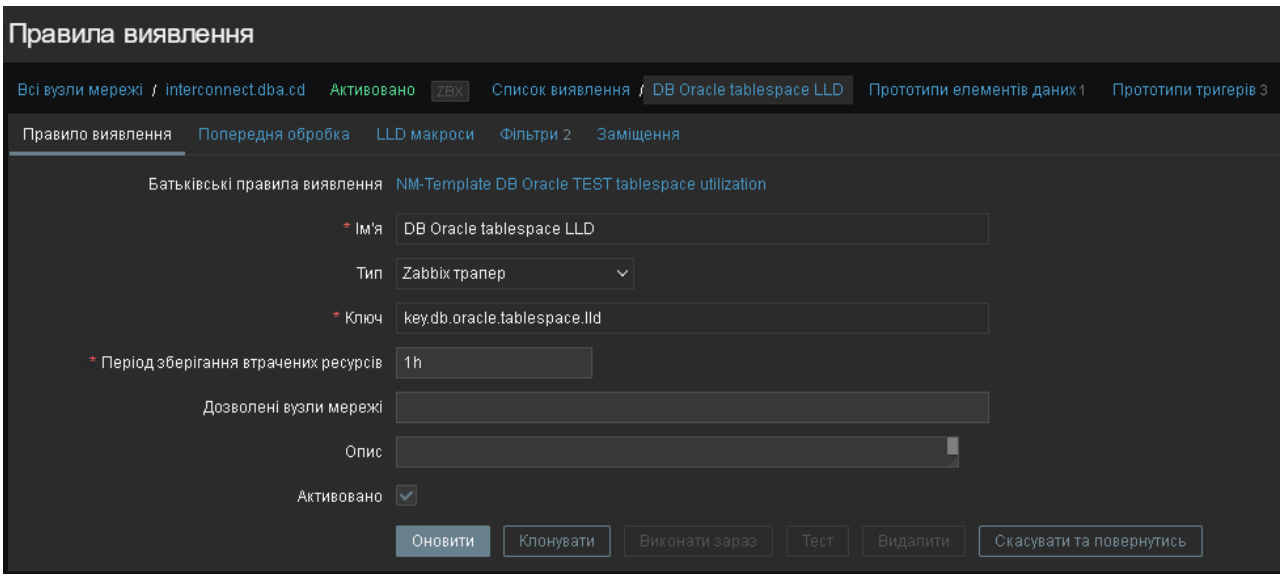


Рис. 2.28 Створення правила виявлення

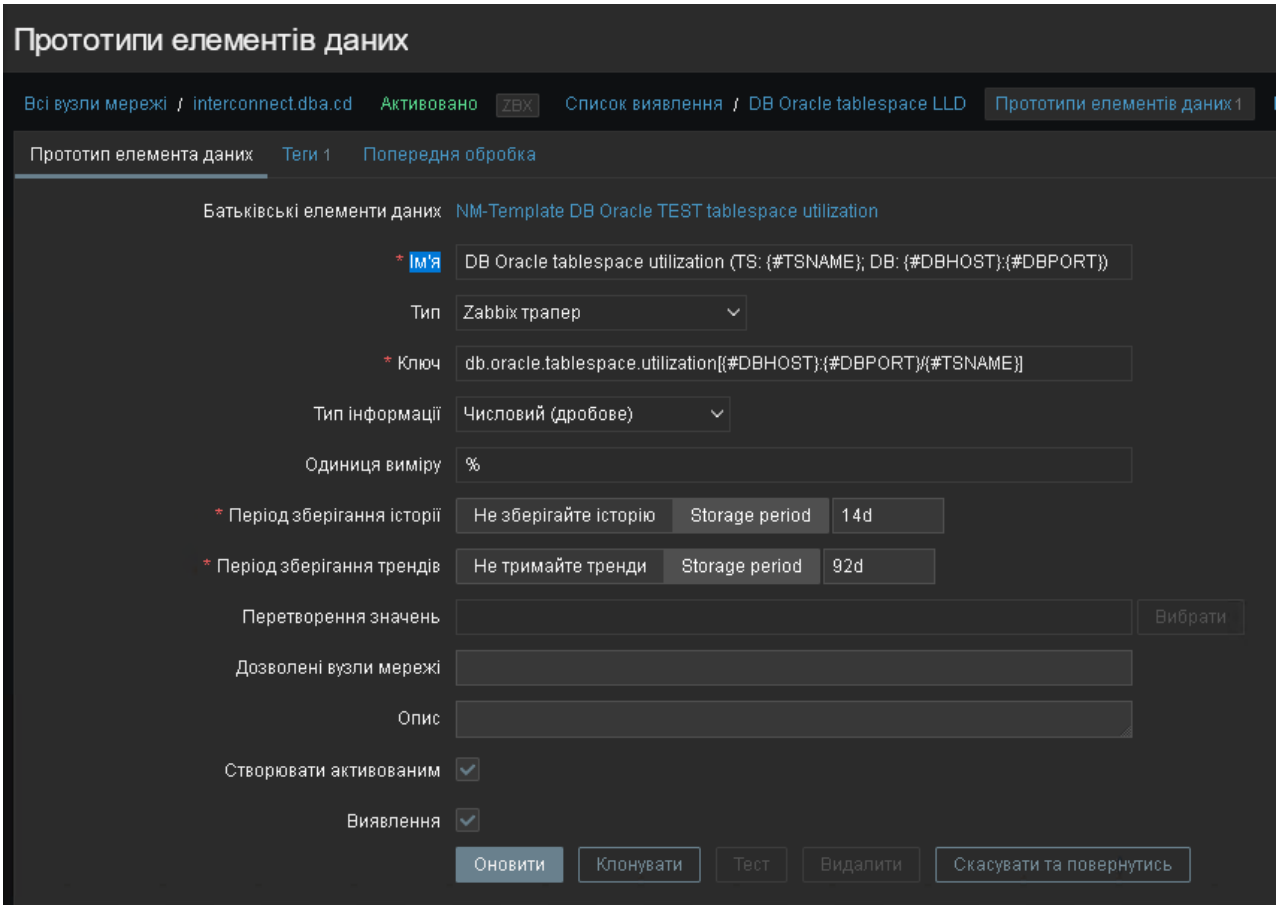


Рис. 2.29 Створення прототипу елементів даних

Далі, нам треба організувати отримання даних від БД Oracle та відправку їх на Zabbix сервер. Для цього створюємо скрипт interconnect_util_check.sh, який буде запускатися кожні 10 хвилин за допомогою стандартних systemd таймерів Linux. Скрипт містить наступний код:

```
#!/usr/bin/bash
# Zabbix Server/Proxy connection parameters.
zabbixserver='127.0.0.1'
zabbixserverport='10051'
zabbixhost='interconnect.dba.cd'
# Database connection parameters.
db_host='172.20.35.5'
db_port='1521'
# Working directory and files.
```

```
workdir=/usr/lib/zabbix/checkscripts/interconnect
pidfile=${workdir}/interconnect_util.pid
sqlquery=${workdir}/interconnect_util_query.sh
itemlist=${workdir}/interconnect_util_item.list
#Script PID-file check.
if [ -f $pidfile ]
then
echo -e "\n$(date +%F" "%T) ERROR: Script PID-file exist. Check if previously started script is still running."
exit 1
else
umask 0077; echo $$ > $pidfile; umask 0033
   # Clean up zabbix_sender 'itemlist' before start.
   umask 0077; > $itemlist; umask 0033
   eval $sqlquery > $workdir/sql.tmp;
   cat $workdir/sql.tmp | \\
   while read line
   do
        # Clean variables before processing of the each line.
        for var in tablespace tablespace_util
        do
             unset $var
        done

        # Define 'tablespace' and 'tablespace_util' variables.
        tablespace=${line%|*}
        tablespace_util=${line#*|}
        # Put data into 'itemlist'.
        if [ -n "$zabbixhost" -a -n "$db_host" -a -n "$db_port"  -a -n "$tablespace" -a -n "$tablespace_util" ]
```

```
        then
            echo "\\"$zabbixhost\\" key.db.oracle.tablespace.lld
[{\\"{#DBHOST}\\":\\"$db_host\\",\\"{#DBPORT}\\":\\"$db_port\\",\\"{#TSNAME}\\
":\\"$tablespace\\"}]" >> \\
                $itemlist
            echo "\\"$zabbixhost\\"
db.oracle.tablespace.utilization[${db_host}:${db_port}/${tablespace}]
0$tablespace_util" >> \\
                $itemlist
        else
            echo -e "\\n$(date +%F" "%T) ERROR: Zabbix sender command option
wrong or missed."
        fi
    done
    # Send items from 'itemlist'.
    zabbix_sender_command="zabbix_sender -z $zabbixserver -p $zabbixserverport -i
$itemlist"
    if [ -n "$zabbixserver" -a -n "$zabbixserverport"  ]
    then
        echo -e "\\n$(date +%F" "%T) $zabbix_sender_command"
        eval $zabbix_sender_command
    else
        echo -e "\\n$(date +%F" "%T) ERROR: Zabbix sender command option wrong
or missed."
    fi
# End of the script. Delete script PID-file.
rm -f $pidfile
fi
exit 0
```

На початку скрипту ми задаємо параметри з’єднання до Zabbix серверу та БД Oracle. Далі визначаємо робочі файли та директорії та виконуємо скрипт interconnect_util_query.sh, який за допомогою команди sqlplus з’єднується з БД та отримує ім’я таблиці та процент утилізації. Ця інформація, розділена символом “|” потрапляє у файл sql.tmp у вигляди таких рядків:

ERM_SYSTEM|46.8

ERM_SYSAUX|59.8

…….

Наступним кроком, формується файл interconnect_util_item.list який містить інформацію у двох записах для кожної таблиці. Перший запис: JSON-об’єкт для Zabbix LLD (Low-Level Discovery), що містить дані про хост бази даних, порт і ім'я tablespace. Другий запис: дані про утилізацію tablespace у певному форматі.
Записи для перших двох таблиць виглядають наступним чином:
"interconnect.dba.cd" key.db.oracle.tablespace.lld
[{"{#DBHOST}":"172.20.35.5","{#DBPORT}":"1521","{#TSNAME}":"ERM_SYSTEM"}]
"interconnect.dba.cd"
db.oracle.tablespace.utilization[172.20.35.5:1521/ERM_SYSTEM] 046.8
"interconnect.dba.cd" key.db.oracle.tablespace.lld
[{"{#DBHOST}":"172.20.35.5","{#DBPORT}":"1521","{#TSNAME}":"ERM_SYSAUX"}]
"interconnect.dba.cd"
db.oracle.tablespace.utilization[172.20.35.5:1521/ERM_SYSAUX] 059.8

За допомогою утиліти zabbix_sender дані з файлу interconnect_util_item.list надсилаються у Zabbix Server або Zabbix Proxy.

Скрипт interconnect_util_query.sh, який викликається описаним вище скриптом interconnect_util_check.sh, за допомогою команди sqlplus з’єднується з БД та отримує ім’я таблиці та процент утилізації.

Скрипт містить наступний код:

```
#!/usr/bin/bash
```

```
# Database connection parameters.
db_user='zabbix'
db_password='Test#$Pa$$wd'
db_host='172.20.35.5'
db_port='1521'
db_service_name='ERM'
# Connect to database using 'sqlplus' and select tablespace_name and space utilization in % from table(sys.check_space().
timeout 50 sqlplus -S << EOF
${db_user}/${db_password}@${db_host}:${db_port}/${db_service_name}
select ds.tablespace_name ||'|'|| to_char(decode(total,0,0,
round(used/total*100,2)),'FM00D00') pct from (select tablespace_name, sum(bytes) used from dba_segments group by tablespace_name) ds, (select tablespace_name, sum(decode(maxbytes,0,bytes,maxbytes)) total from dba_data_files group by tablespace_name) ddf where ds.tablespace_name = ddf.tablespace_name;
EOF
```

Таким чином, кожні 10 хвилин у Zabbix будут потрапляти дані про назву таблиць та їх утилізацію. На Рис. 2.30 бачимо створені елементи даних у відповідному розділі. Унікальне ім’я елементу даних складається з назви списку виявлення (DB Oracle tablespace LLD) та назви прототипа елементу даних – ”DB Oracle tablespace utilization” та назви таблиці, хосту БД та порту.

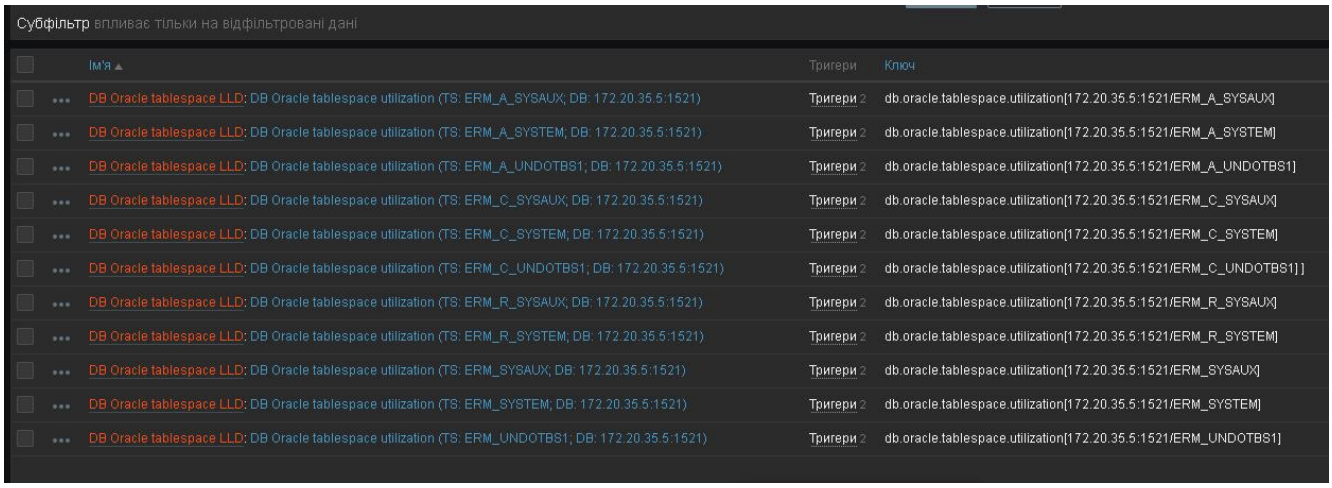


Рис. 2.30 Елементи даних з утилізації БД

У розділі “Останні дані” на Рис. 2.31 бачимо створені елементи даних з фактичними даними утилізації та часом останньої перевірки.

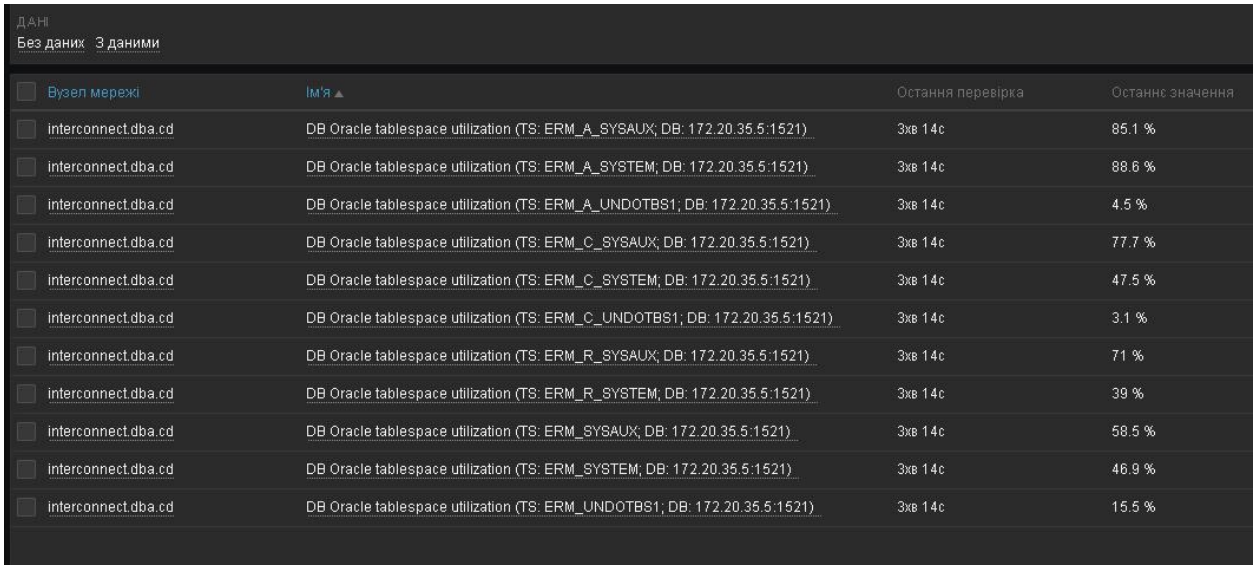


Рис. 2.31 Останні дані з утилізації БД

Таким чином, перевіривши, що останні дані по метрикам утилізації БД надходять, робимо висновки, що моніторинг налаштовано вірно.

Для оповіщень створюємо прототип тригерів зображений на Рис. 2.32, який буде мати важливість “Критична” та спрацьовувати, коли останнє значення метрики буде перевищувати значення задане у макросі {$DB_ORACLE_TABLESPACE_FREE_SPACE_CRITICAL}. Прототип тригера автоматично буде додано до елементів даних (Рис. 2.30) після їх створення.

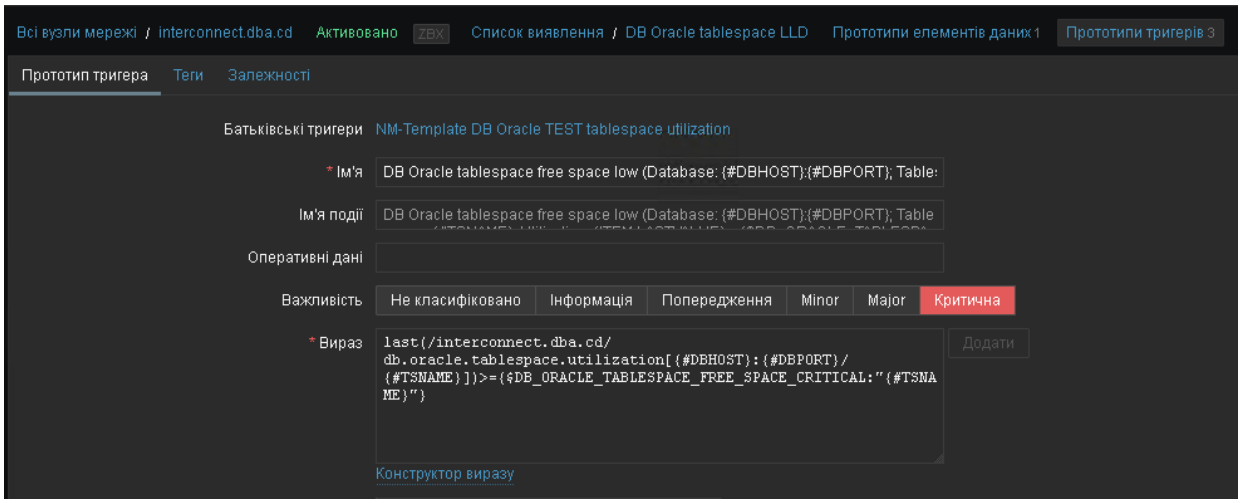


Рис. 2.32 Створення прототипу тригерів

2.3 Огляд моніторингу по протоколу SNMP

Для комутатору interconnect.rtr.cd типу Cisco WS-C3650 треба організувати моніторинг по протоколу SNMP. Особливість такого моніторингу полягає в тому, що на пристрій не встановлюється Zabbix агент, а вся взаємодія з Zabbix сервером виконується по відкритому SNMP протоколу. Агенти віддаленого моніторингу прослуховують порт 161 (UDP) на предмет запитів від системи моніторингу та повертають відповіді на запити або SNMP- пастки (SNMP-traps) по порту 162 (UDP).

Створюємо вузол мережі аналогічно двом попереднім, але тип інтерфейсу вказуємо SNMP, версію SNMP – SNMPv2 та порт – 161 (Рис. 2.33). Макросом {$SNMP_COMMUNITY} вказуємо так звану community string, що виконує роль ідентифікатора відправника.

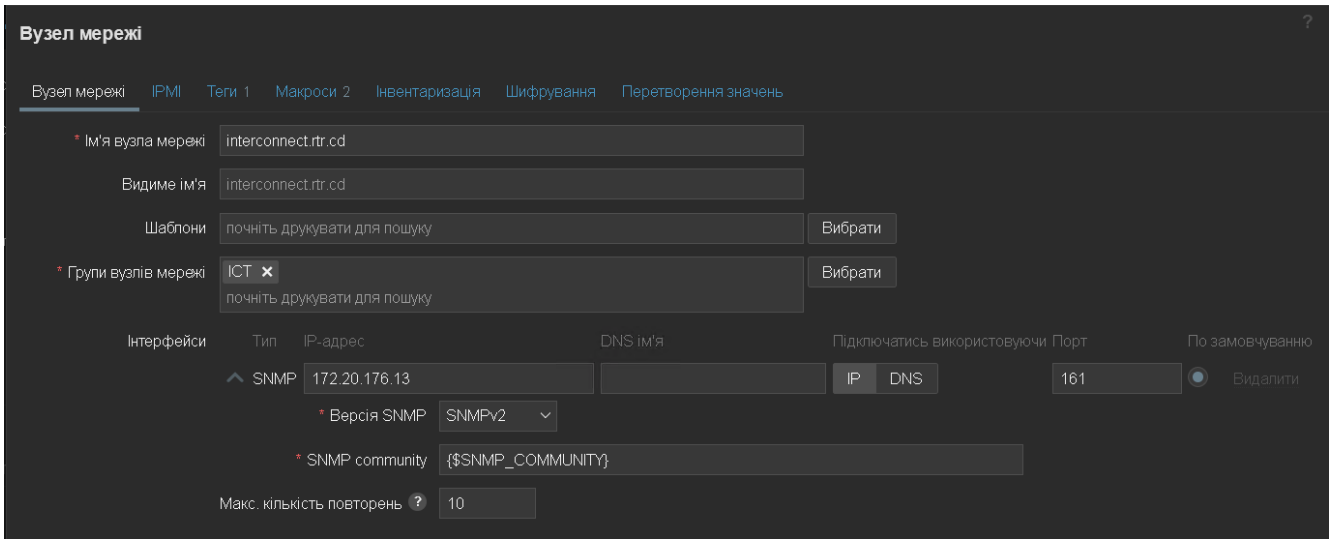


Рис. 2.33 Створення вузла з SNMP інтерфейсом у системі Zabbix

Основні метрики, які будуть встановлюватись на моніторинг по протоколу SNMP наведені у Табл. 2.2

Таблиця 2.2

Основні метрики для моніторингу комутатору по протоколу SNMP

Назва метрики	Опис технічного параметру	Одиниця виміру	Пороги та критичність
Стан портів комутатора	Виявлення портів комутатора та їх робочого стану	число	Критична: = 2 (порт не працює)
Обсяг вхідного трафіку на портах	Завантаження вхідного трафіку порівняно з максимальною пропускною здатністю порту (для кожного порту).	%	Критична: > 80 %
Обсяг вихідного трафіку на портах	Завантаження вихідного трафіку порівняно з максимальною пропускною здатністю порту (для кожного порту).	%	Критична: > 80 %

Щоб отримати дані для метрики “Стан портів комутатора” створюємо правило виявлення (Рис. 2.34). OID'и для виявлення додаються в поле SNMP OID у наступному форматі: discovery[{#МАКРОС1}, oid1, {#МАКРОС2}, oid2, …,]. У нашому випадку це буде запит: discovery[{#PORT},IF-MIB::ifDescr,{#PORTSTATUS},IF-MIB::ifAdminStatus]. Отримувати за таким правилом будемо масив значень вигляду: [{"{#SNMPINDEX}":"1","{#PORT}":"GigabitEthernet0/0","{#PORTSTATUS}":"2"},{"{#SNMPINDEX}":"2","{#PORT}":"GigabitEthernet1/0/2","{#PORTSTATUS}":"1"},…].

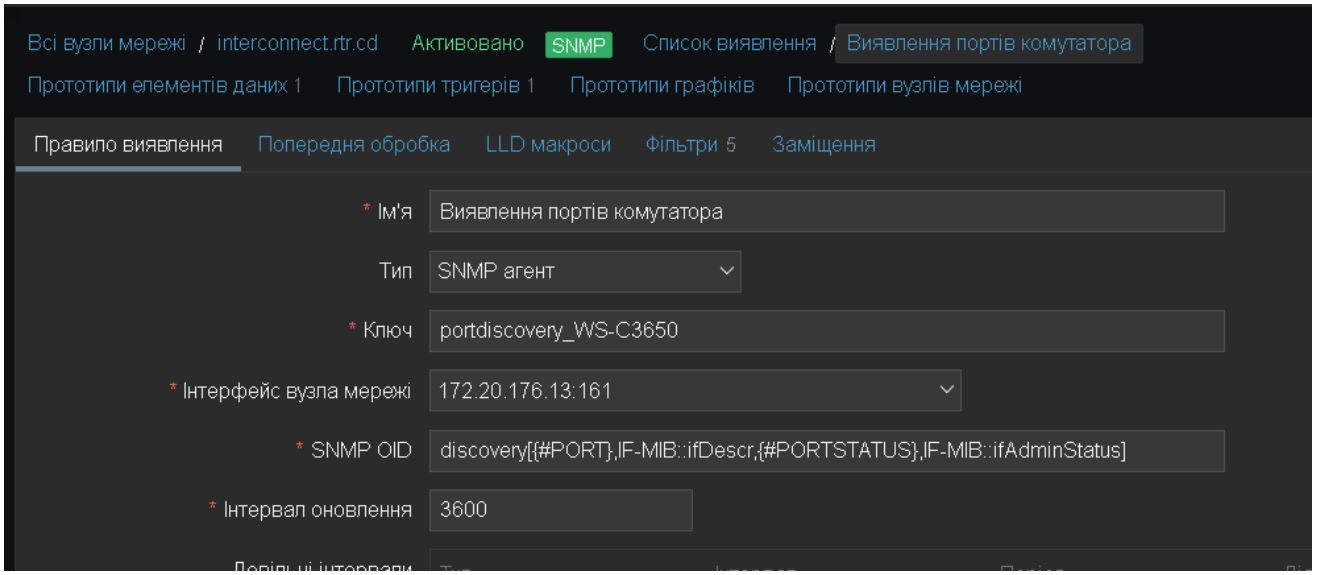


Рис. 2.34 Правило виявлення портів комутатора

Інтервал оновлення правила виявлення інформації по портам вказуємо рівним 3600 секунд.

Прототипи елементів даних з виявлення портів комутатора наведено на Рис 2.35. Інтервал оновлення інформації для елементів даних вказуємо рівним 300 секунд.

У OID SNMP запитах тексти, як IF-MIB::ifAdminStatus та IF-MIB::ifDescr є частиною MIB (Management Information Base) та фактично замінюють цифрову частину OID. Нижче наведені приклади таких текстів з описом та OID-ами, які вони заміщують:

- IF-MIB::ifDescr — 1.3.6.1.2.1.2.2.1.2 (текстовий опис інтерфейсу);
- IF-MIB::ifAdminStatus — 1.3.6.1.2.1.2.2.1.7 (статус інтерфейсу, який показує чи увімкнутий цей інтерфейс адміністратором чи ні);
- IF-MIB::ifOperStatus — 1.3.6.1.2.1.2.2.1.8 (статус роботи інтерфейсу).

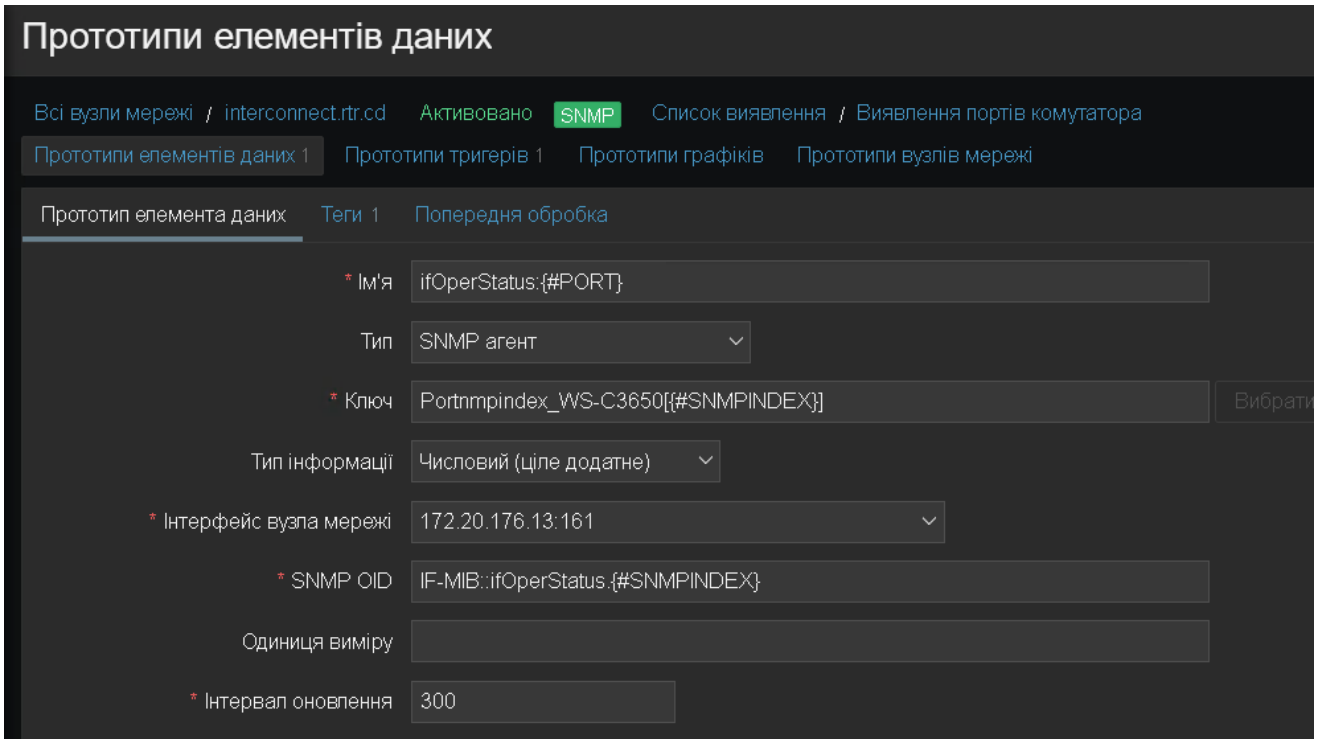


Рис. 2.35 Прототип елементів даних з виявлення портів комутатора

Як ifAdminStatus так і ifOperStatus можуть приймати значення 1 (якщо порт увімкнутий або працює) або 2 (якщо порт не працює або вимкнений примусово).

Прототипи тригеру з виявлення портів комутатора наведено на Рис 2.36.

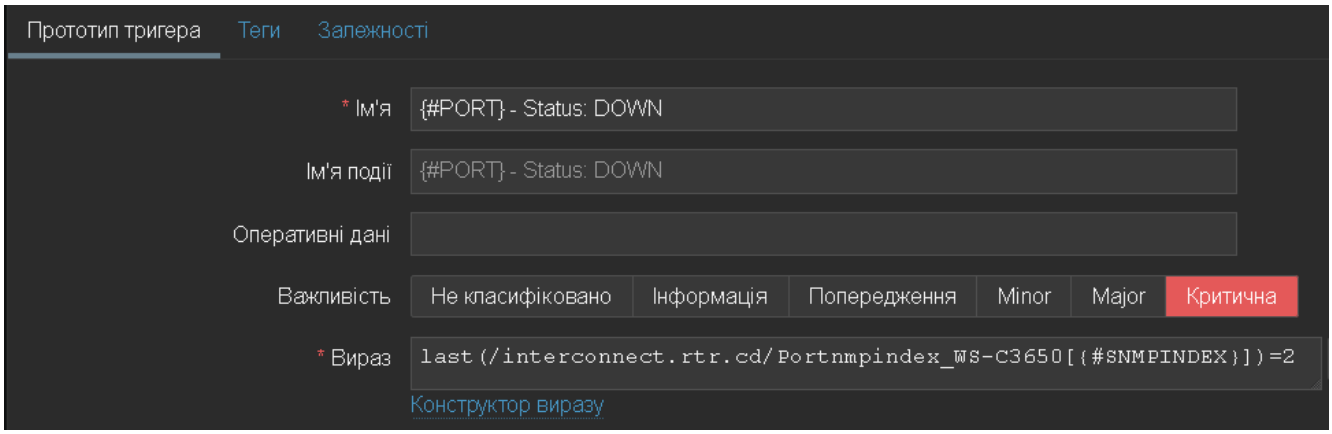


Рис. 2.36 Прототип тригеру з виявлення портів комутатора

Обсяг вхідного та вихідного трафіку на портах у Табл. 2.2 розраховується за допомогою правила виявлення та додаткових прототипів елементів даних. Аналогічно правилу виявлення “Стан портів комутатора” (Рис. 2.34), створюємо нове правило “Виявлення_ ifDescr” (Рис. 2.37).

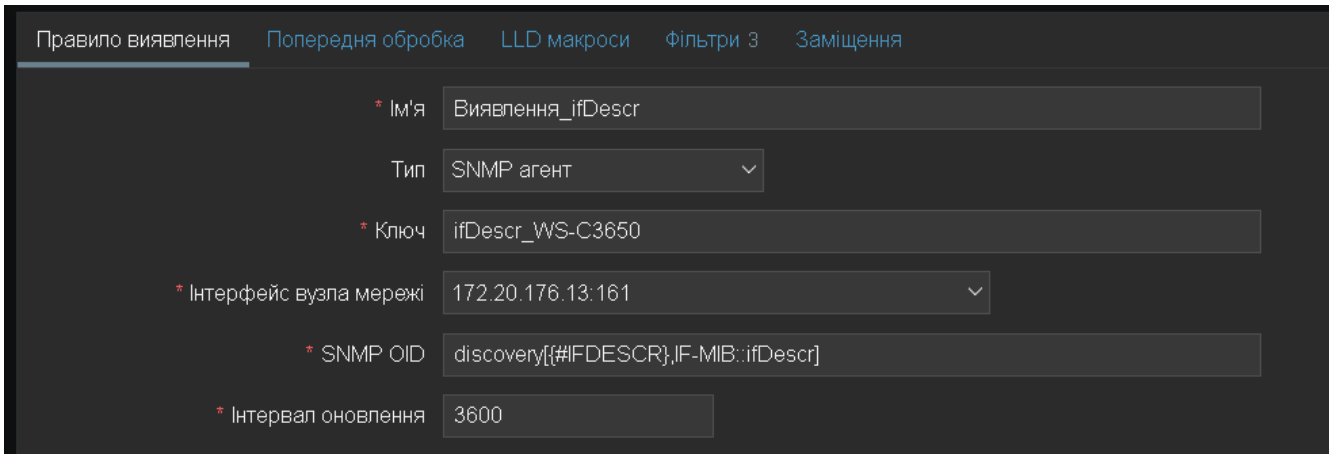


Рис. 2.37 Правило виявлення для обчислення обсягу вхідного трафіку на портах комутатора

SNMP OID задаємо у вигляді: “discovery[{#IFDESCR},IF-MIB::ifDescr]”. Після виконання правила виявлення кожну годину, система Zabbix буде отримувати масив значень вигляду:

[{"{#SNMPINDEX}":"1","{#IFDESCR}":"GigabitEthernet0/0"},{"{#SNMPINDEX}"

:"2","{#IFDESCR}":"Null0"},{"{#SNMPINDEX}":"3","{#IFDESCR}":"unrouted VLAN 1"},…]

На комутаторі, який ставиться на моніторинг, виявляється близько 400 портів. Частина з них може були адміністративно заблоковані, частина – нам не цікава тому, що не несе інформації про продуктивний трафік. Тому, у правилі виявлення може бути застосована фільтрація результатів пошуку (Рис. 2.38). У даному випадку нам цікаво моніторити порти, які включають назву “GigabitEthernet” та закінчуються на цифри “1/1” “1/2”.

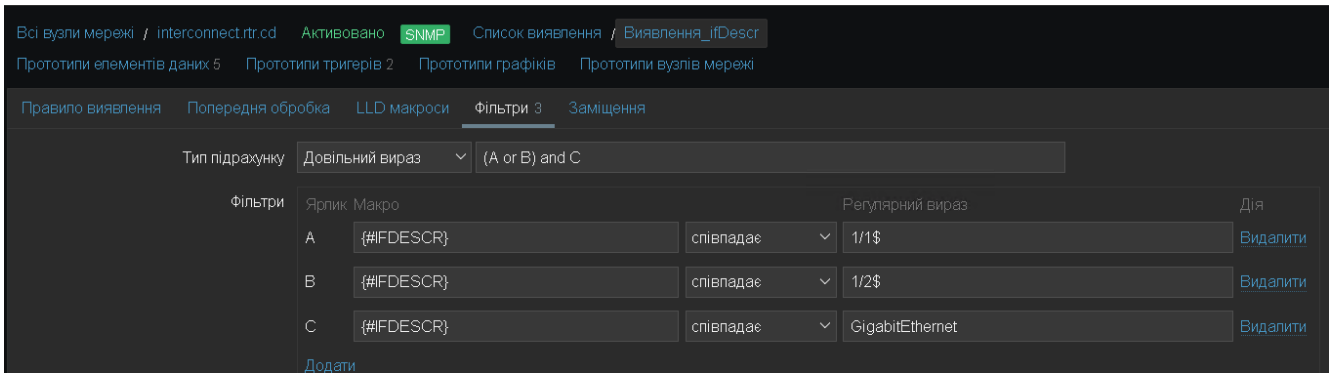


Рис. 2.38 Фільтрація небажаних портів у правилі виявлення портів

Для розрахунку обсягу вхідного та вихідного трафіку у процентах нам знадобиться три елементи даних:

- Лічильник кількості отриманих байтів (октетів) через певний інтерфейс;
- Лічильник кількості переданих байтів (октетів) через певний інтерфейс;
- Максимально можлива швидкість інтерфейсу.

На Рис. 2.39 зображено приклад прототипу елементу даних, який підраховує кількість отриманих байтів (октетів) через певний інтерфейс. SNMP OID задається у вигляді тексту “IF-MIB::ifInOctets.{#SNMPINDEX}”, де ifInOctets перетворюється Zabbix на 1.3.6.1.2.1.2.2.1.10.

Для підрахунку кількості переданих байтів використовується аналогічний прототип елементу даних з ім’ям “Interface Out {#FDESCR}” та SNMP OID-ом, який задається у вигляді тексту “IF-MIB::ifOutOctets.{#SNMPINDEX}”, де ifOutOctets дорівнює вже іншому MIB - 1.3.6.1.2.1.2.2.1.16.

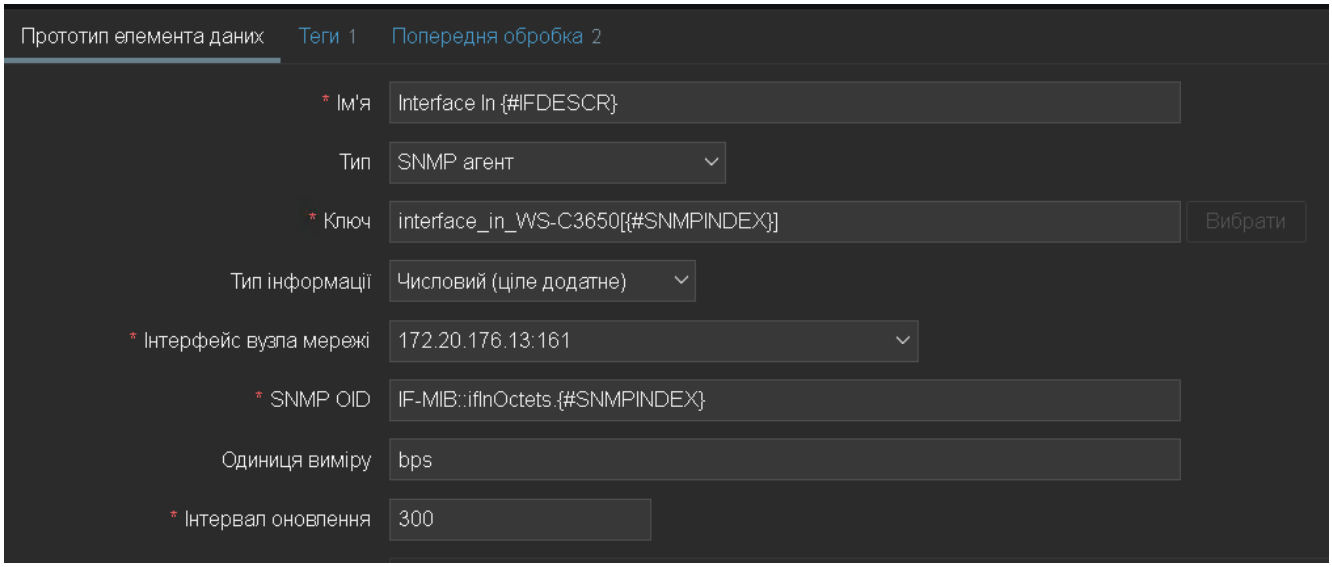


Рис. 2.39 Прототип елементу даних для підрахунку отриманих байтів

На вкладці “Попередня обробка” (Рис. 2.40) виконується обчислення зміни значення (різниця між поточним та попереднім значеннями) – швидкість в секунду. Наступним кроком результат множиться на 8, щоб перетворити октети на біти.

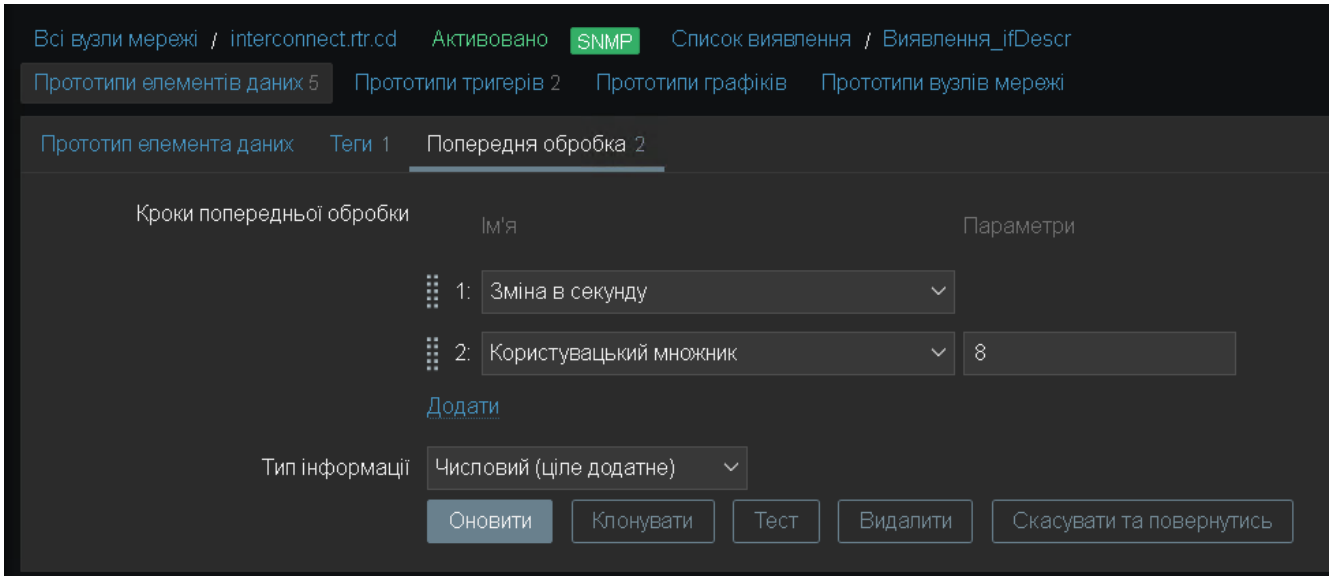


Рис. 2.40 Попередня обробка прототипів елементів даних для підрахунку отриманих та переданих бітів

Максимально можлива швидкість інтерфейсу отримується так само, як і два попередніх прототипу даних (Рис 2.39), але присвоюємо йому ім’я “ if Speed {#IFDESCR}” та SNMP OID, який задається у вигляді тексту “ IF-

MIB::ifSpeed.{#SNMPINDEX}. Текст ifSpeed дорівнює OID зі значенням “1.3.6.1.2.1.2.2.1.5”.

Прототипи елементів даних для підрахунку отриманих та переданих бітів наведені на Рис 2.41.

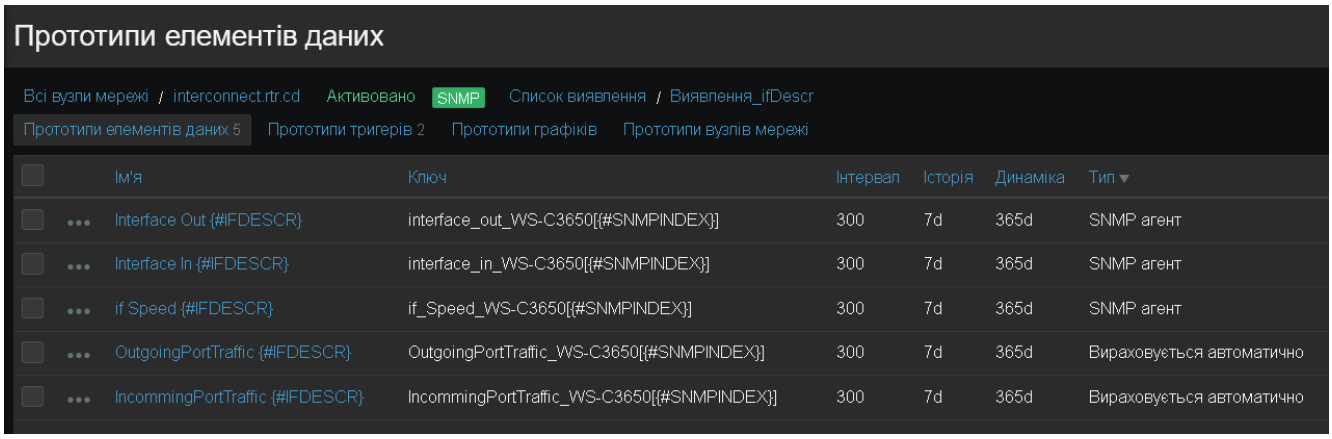


Рис. 2.41 Прототипи елементів даних для підрахунку отриманих та переданих бітів

Два останніх елемента даних на Рис. 2.41 вираховуються автоматично і дорівнюють швидкості бітів за секунду, які на даний момент пропускає інтерфейс поділене на максимально можливу швидкість інтерфейсу у бітах за секунду. Приклад прототипа елементів даних для підрахунку обсягу вихідного трафіку на портах у % наведено на Рис. 2.42.

Створюємо прототип тригерів (Рис. 2.43) для перевищення обсягу вихідного трафіку на 80%. Аналогічно, для прототипа тригерів по перевищенню вхідного трафіку на 80%, у виразі змінюється тільки метрика “ OutgoingPortTraffic_WS-C3650[{#SNMPINDEX}]” на метрику “ IncommingPortTraffic_WS-C3650[{#SNMPINDEX}]”.

Таким чином, ми побудували моніторинг інтерфейсів комутатора, що дає змогу динамічно відслідковувати стан портів та критичне перевищення порогів утилізації вхідного та вихідного трафіку на портах.

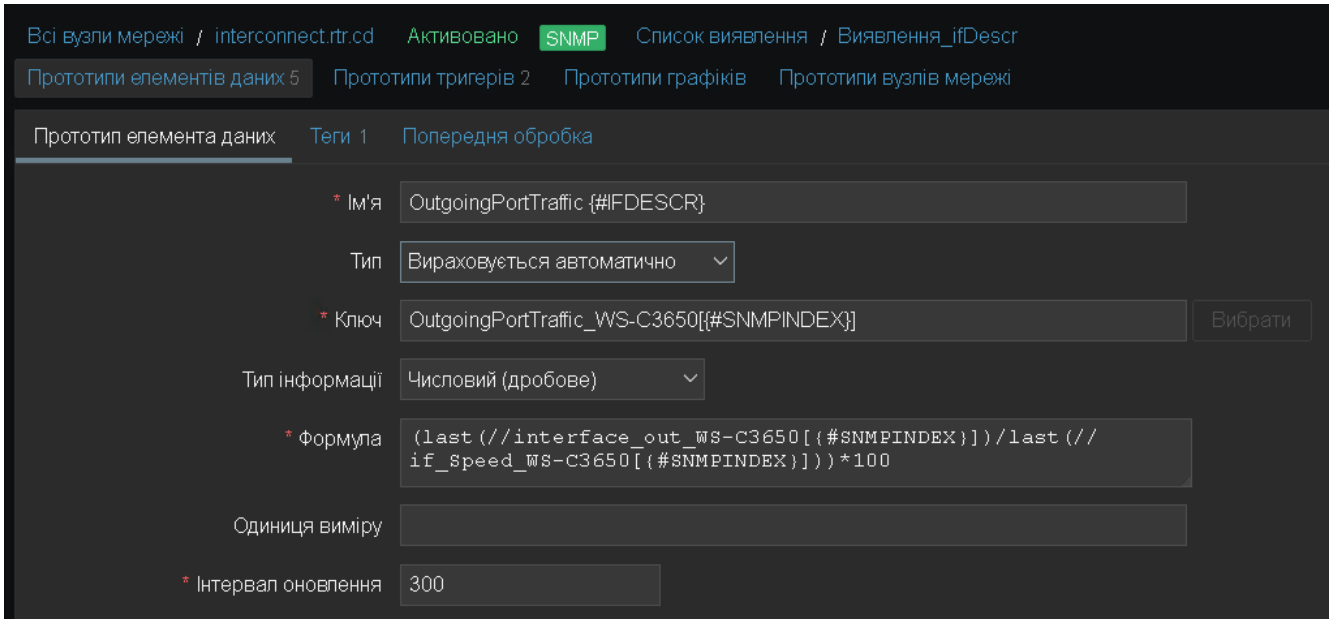


Рис. 2.42 Прототип елементів даних для підрахунку обсягу вихідного трафіку на портах у %

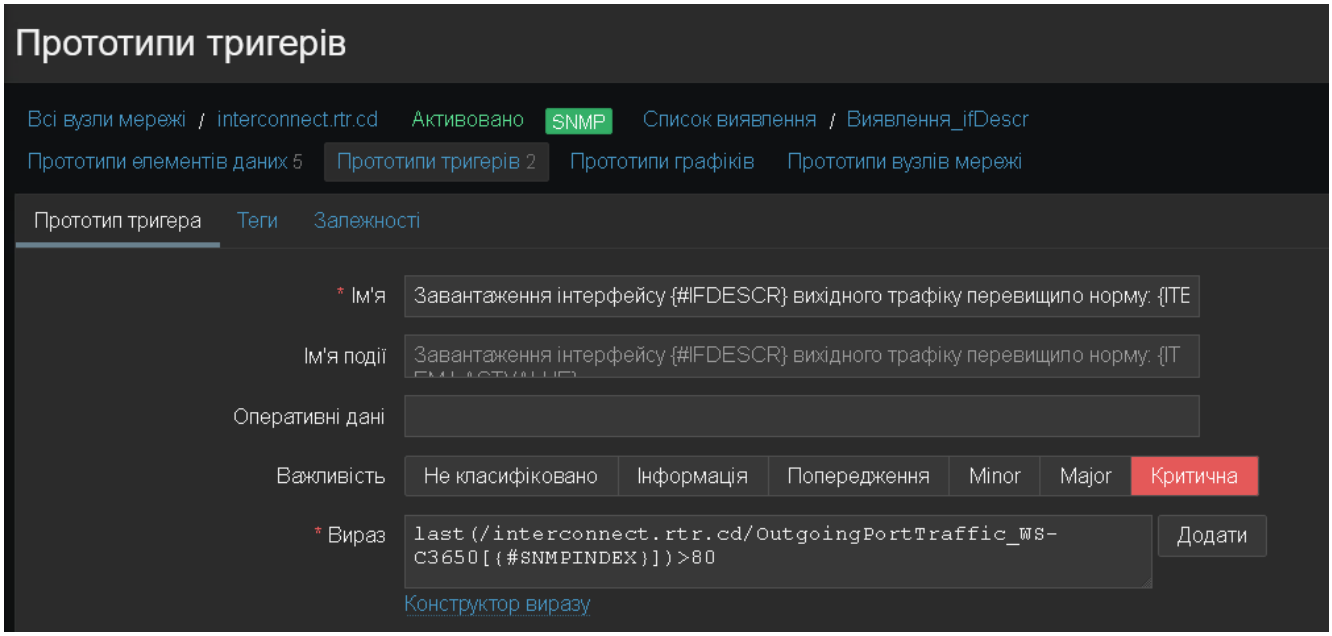


Рис. 2.43 Прототип тригерів для перевищення обсягу вихідного трафіку на портах у %

Останнім кроком необхідно встановити автоматичні оповіщення у відповідь на спрацювання тригерів (відправка повідомлення на електронну пошту або надсилання повідомлень). Для цього у розділі “Оповіщення” – “Дії” створюємо новий запис “Дії тригерів” (Рис. 2.44).

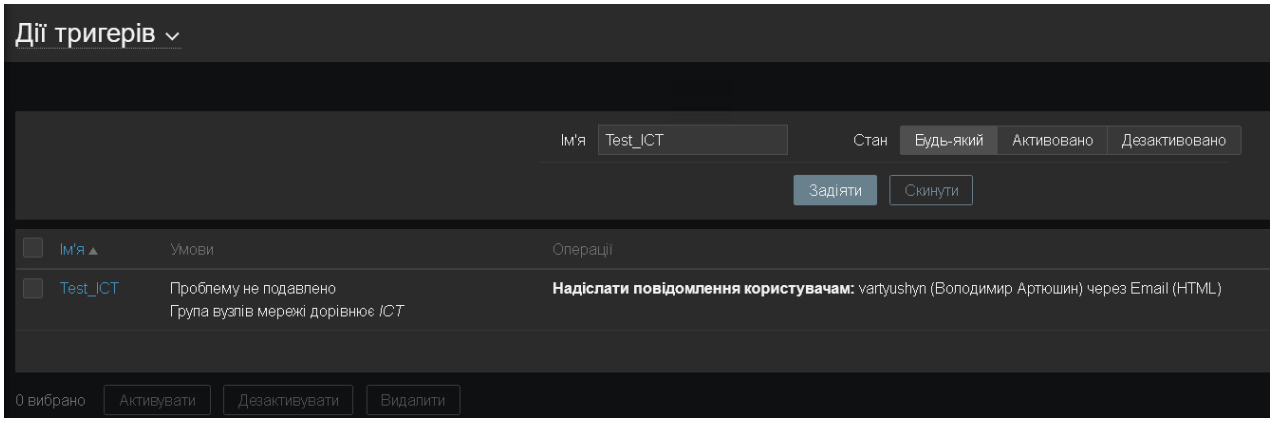


Рис. 2.44. Встановлення автоматичних оповіщень у відповідь на спрацювання тригерів

Створені у даній кваліфікаційній магістерській роботі вузли мережі були об’єднані у групу вузлів “ІСТ”. Тому, інформацію про всі проблеми будемо надсилати на користувача у вигладі електронної пошти.

Підсумовуючи все вищесказане, можна записати алгоритм встановлення на моніторинг нових мережевих елементів мобільного оператора (Рис 2.45).

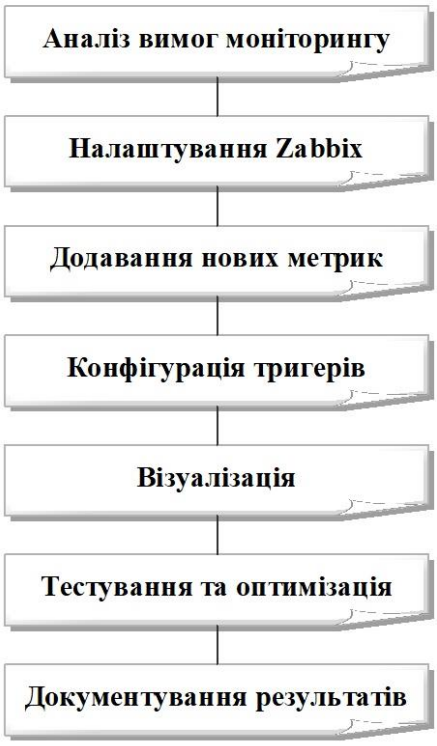


Рис. 2.45. Алгоритм встановлення на моніторинг нових мережевих елементів мобільного оператора

Щоб налаштувати моніторинг мобільного оператора за допомогою системи Zabbix, необхідно врахувати архітектуру мережі оператора, визначитися, що буде моніторитися (хости, сервіси, мережа та інше) та якими методами або інструментами буде виконуватися моніторинг.

Покроковий алгоритм можна описати у вигляді таких кроків:

1. Аналіз вимог моніторингу:
   - визначення, що потрібно моніторити (обладнання або сервіси);
   - вибір ключових метрик, які потрібно моніторити.
2. Налаштування Zabbix:
   - встановлення Zabbix агентів вузлах, які потрібно моніторити, та їх конфігурація;
   - забезпечення доступності у мережі між Zabbix сервером або Zabbix Proxy сервером та вузлами, які моніторяться;
   - додавання нових вузлів (hosts) у веб-інтерфейсі Zabbix.
3. Додавання нових ключових метрик для вузлів, які встановлюються на моніторинг.
4. Конфігурація тригерів, яка включає:
   - визначення умов спрацьовування у вигляді логічного виразу, який задає умови, коли тригер має активуватись;
   - налаштування рівня важливості (інформація, попередження, minor (неважлива), major (важлива), критична).
   - встановлення автоматичних дій у відповідь на спрацювання тригеру (відправка повідомлення на електронну пошту або надсилання повідомлень).
5. Візуалізація, яка включає створення дашбордів або інформаційних панелей для відображення метрик та інших графічних результатів.
6. Тестування та оптимізація:
   - тестування спрацювання тригерів за умови перевищення показників або інших заданих умов;
   - оптимізація тригерів, для мінімізації хибних сповіщень (наприклад,

визначення мінімального часу тривалості аварії для спрацювання тригеру).

7. Документування результатів, яке включає опис метрик та умов спрацювання тригерів а також, інструкції для чергового персоналу при винекненні аварійних ситуацій.

3. ДОСЛІДЖЕННЯ РЕЗУЛЬТАТІВ РОБОТИ СИСТЕМИ МОНІТОРИНГУ

3.1 Огляд інформаційної системи моніторингу

У розділі 2 була розроблена інформаційна система моніторингу мобільного оператора на базі системи Zabbix. Для прикладу, була розглянута система білінгу міжоператорського трафіку, яка складається з двох серверів на базі ОС Linux interconnect.nss.cd та interconnect.dba.cd. На останньому з них встановлена БД Oracle. У склад системи також входить мережевий комутатор interconnect.rtr.cd, який моніториться по протоколу SNMP. Ми об'єднали два сервери та комутатор у групу вузлів “ICT” (Рис. 3.1).

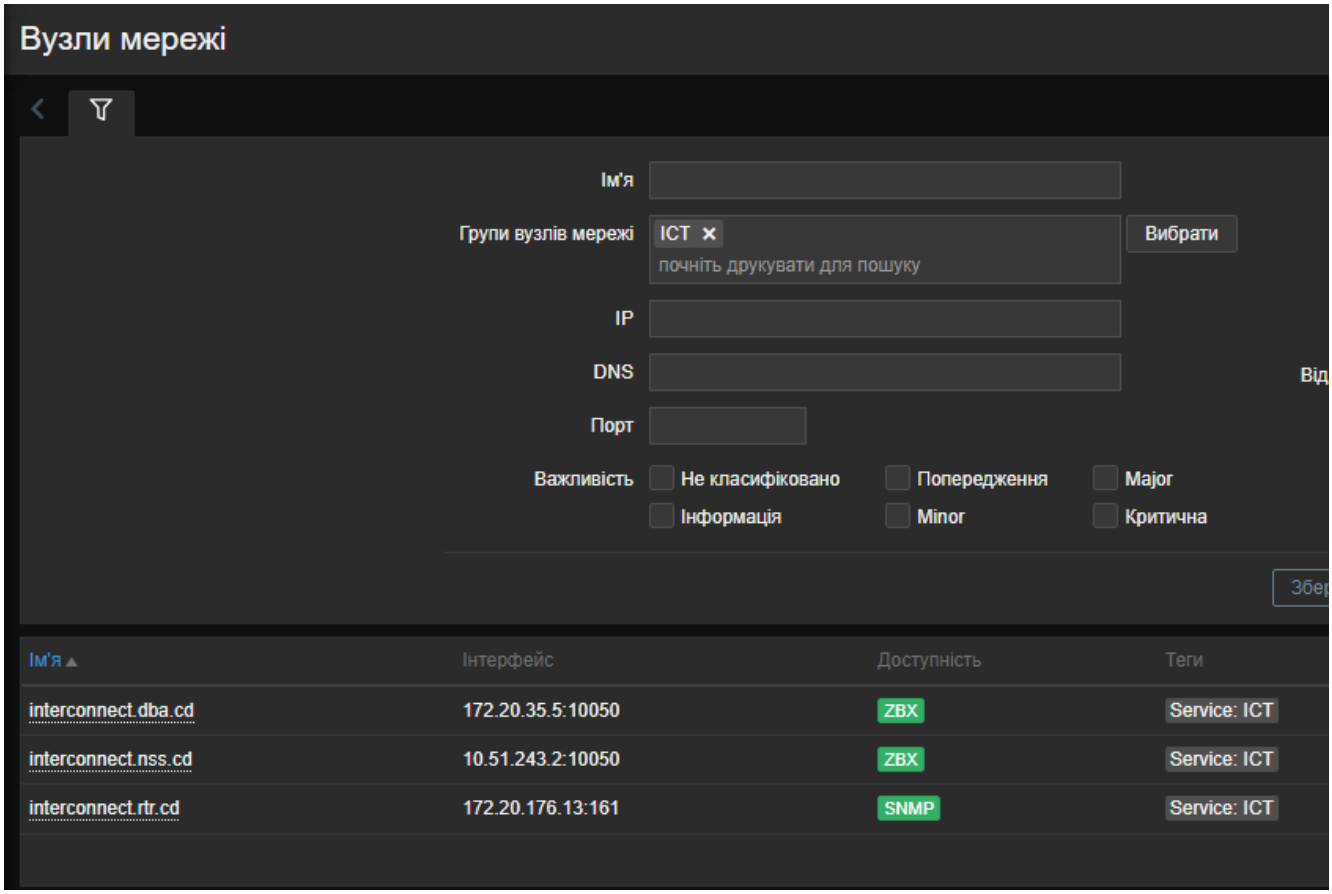


Рис. 3.1 Об’єднання вузлів у групу

Таке об’єднання спрощує пошук вузла, коли у оператора є багато систем і не завжди очевидно, до якої системи він належить. Також, зазвичай, за різні

системи відповідають різні спеціалісти або групи спеціалістів. Тому, набагато простіше організувати інформування групи користувачів про проблемі тієї чи іншої системи.

Як було описано у алгоритмі по налаштуванню моніторингу на Рис. 2.45, для відображення метрик та інших графічних результатів розробником виконується візуалізація у вигляді дашбордів або інформаційних панелей. Було вирішено зробити інформаційну панель з чотирма вкладками. Для полегшення візуалізації (треба щоб панелі були видимі на одному екрані без необхідності прокрутки) дві перші вкладки будуть відображати основні метрики ОС Linux з Табл. 2.1. На Рис 3.2 наведена перша вкладка інформаційній панелі, яка відображає:

- Утилізацію пам'яті (Memory utilization, %);
- Утилізація файлу підкачки (SWAP) в % (графік побудовано для вільного місця у файлу підкачки - Free SWAP space, %);
- Утилізація процесору (CPU total utilization in %);
- Середне навантаженя на процесор (1, 5, 15 хвилинний інтервал) - CPU load avarage in %.

Рис. 3.2 Вкладка №1 з основними метриками ОС Linux

На Рис 3.3 наведена друга вкладка інформаційній панелі, яка відображає:

- Використання простору в % (для кожного розділу файлової системи) (Disk space utilization, %);
- Завантаження вхідного та вихідного трафіку порівняно з максимальною пропускною здатністю мережевого інтерфейсу (для кожного мережевого інтерфейсу) – Interface utilization, %;
- Кількість помилок (для кожного мережевого інтерфейсу) – Interface error rate;
- Стан мережевого інтерфейсу (Operational status).

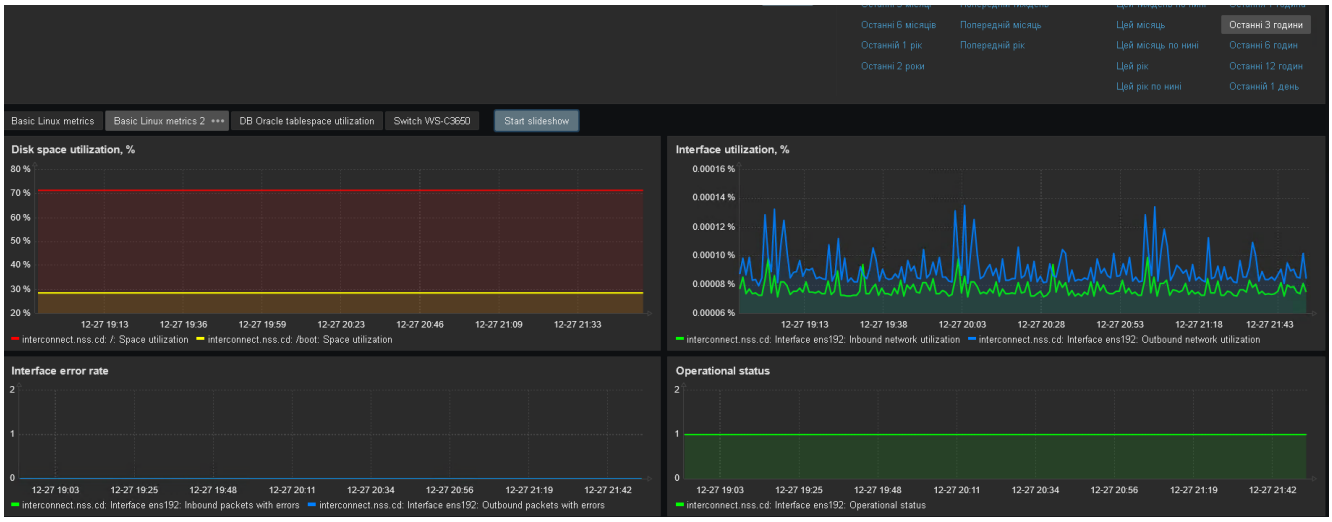


Рис. 3.3 Вкладка №2 з основними метриками ОС Linux

У нашому випадку, мережевий інтерфейс у сервера, який треба моніторити, тільки один – “ens192”, а файлових систем дві – “/” “/boot”.

Третя вкладка на інформаційній панелі (Рис. 3.4) відображає метрики, описані в підрозділі 2.2:

- Утилізація таблиць БД Oracle (DB Oracle tablespace utilization) в %;
- Моніторинг доступності з'єднання з БД (SQL connection status).

На графіку виведені всі 11 таблиць, у яких треба моніторити утилізацію табличного простору. В нашому випадку зберігається наочність інформації, але, якщо таких метрик на один графік буде багато і вони будут збігатися або перетинатися – треба розділяти їх по різним графікам.

Рис. 3.4 Вкладка №3 з метриками утилізації БД Oracle та моніторингу доступності з'єднання з БД

На Рис 3.5 наведена четверта вкладка інформаційної панелі, яка відображає:

- Обсяг вхідного трафіку на портах (Incoming port traffic);
- Обсяг вихідного трафіку на портах (Outgoing port traffic);
- Стан портів комутатора (Interface operational status).

Рис. 3.5 Вкладка №4 з метриками трафіку та станом портів комутатору

Як бачимо на графіках, тільки два інтерфейсу працюють та через них

проходить продуктивний трафік (ifOperStatus=1). Решта два інтерфейсу вимкнено та їх статус роботи дорівнює “2” (ifOperStatus=2).

Таким чином, ми побудували інформаційну панель на якій виконано візуалізацію метрик з двох серверів та комутатору, які були описані у розділі 2. Це дозволяє, у випадку аварії, перевірити наявність метрик, їх значення та вихід за критичні значення. Для зручності фахівців центру моніторингу, під час відображення інформаційної панелі реалізовано механіхм автоматичного перемикання між вкладками з інтервалом 30 секунд. Таким чином, вихід за критичні значення або пропадання метрик можна зафіксувати візуально.

3.2 Тестування інформаційної системи моніторингу

Тестування інформаційної системи моніторингу (Рис 2.45) це важливий етап впровадження нових мережевих елементів або інформаційних систем на мережі мобільного оператора.

Перш за все, перевіряється спрацювання тригерів за умови перевищення показників або інших заданих умов. Це дуже важливий етап тому, що інколи тригер має дуже складний вираз (комбінацію математичних та логічних умов) і потрібно враховувати людський фактор при створенні тригеру.
В ході експлуатації інформаційної системи додатково виникають різні аспекти, які треба враховувати для оптимізація тригерів або для мінімізації хибних сповіщень. Наприклад, інколи виникають поодинокі випадки короткострокового перевищення критичних показників метрики. Але критичним є, наприклад, перевищення таких показників протягом 5-ти хвилин та більше. Тому практично завжди визначається мінімальний час тривалості аварії для спрацювання тригеру.

Виконаємо тестування спрацювання тригеру для метрики утилізація пам'яті (Memory utilization, %). За замовчуванням, спрацювання задаються макросами {$MEMORY.UTIL.MAX.CRIT} (Рис. 2.5) та {$MEMORY.UTIL.MAX.WARN} (Рис. 2.6), які дорівнюють, відповідно, 95% та 90%. Поточне значення метрики з утилізація пам'яті довівнює близько 65%. Тому, зменшуючи пороги задані

макросах, отримаємо аварії у системі для критичної аварії та попередження. На Рис. 3.6 та Рис.3.7 можна побачити спрацювання тригерів критичної аварії та попередження для метрики з утилізації пам’яті у списку тригерів хосту interconnect.nss.cd.

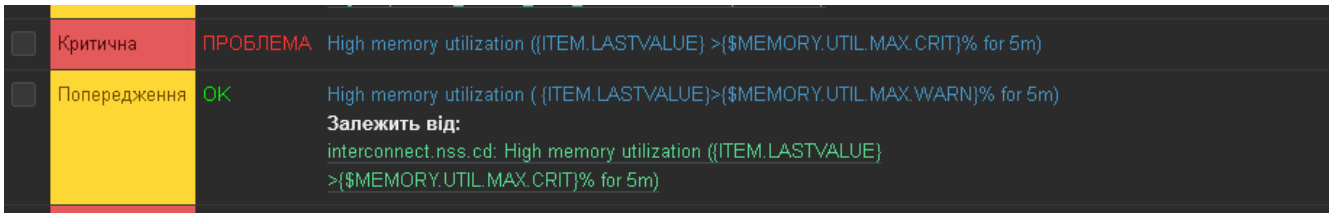


Рис. 3.6 Тестування спрацювання тригеру критичної аварії для метрики з утилізації пам’яті

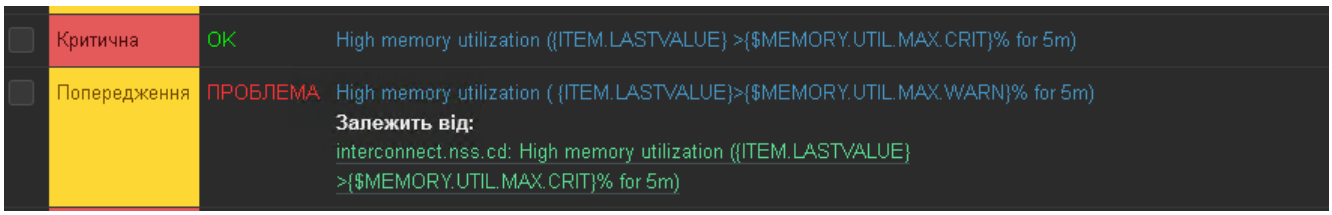


Рис. 3.7 Тестування спрацювання тригеру попередження для метрики з утилізації пам’яті

На Рис. 3.8 бачимо історію проблем (час виникнення проблеми та час відновлення проблеми).

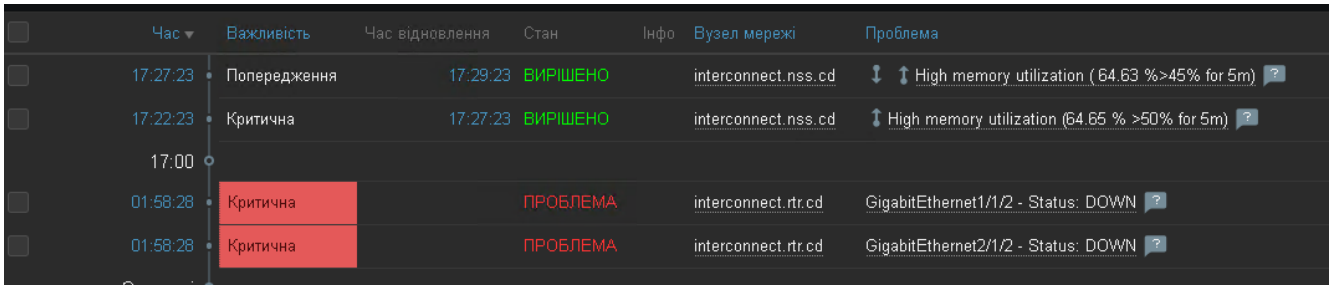


Рис. 3.8 Відображення проблем на вкладці “Проблеми”

Для зручності, права частина екрану з Рис 3.8 перенесена на Рис. 3.9, на якому зображені стовпці з тривалістю проблеми та діями, які були виконані. В

нашому випадку, якщо натиснути на стрілку під написом “Дії”, бачимо, що 2 рази було відправлено електронного листа (початок та закінчення проблеми) для критичної проблеми та 2 рази – для попередження.

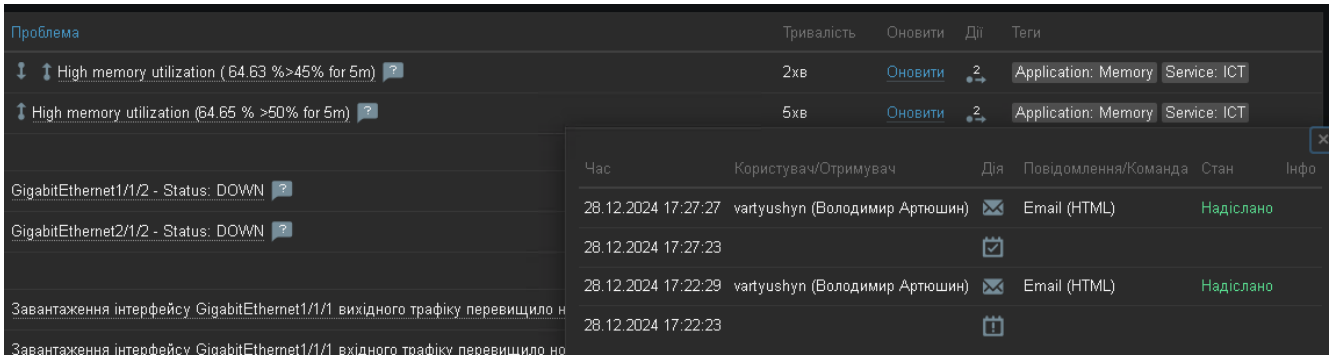


Рис. 3.9 Відображення проблем на вкладці “Проблеми” (продовження).

Проводимо перевірку електронної пошти і знаходимо 4 листа. Два листа було відправлено з приводу початку та закінчення (Resolved) критичної проблеми (Рис. 3.10), та два листа - з приводу початку та закінчення проблеми з важливістю “Попередження” (Рис. 3.11).

Робимо висновки, що сповіщення, які були встановлені на дії тригерів (Рис. 2.44) працюють коректно.

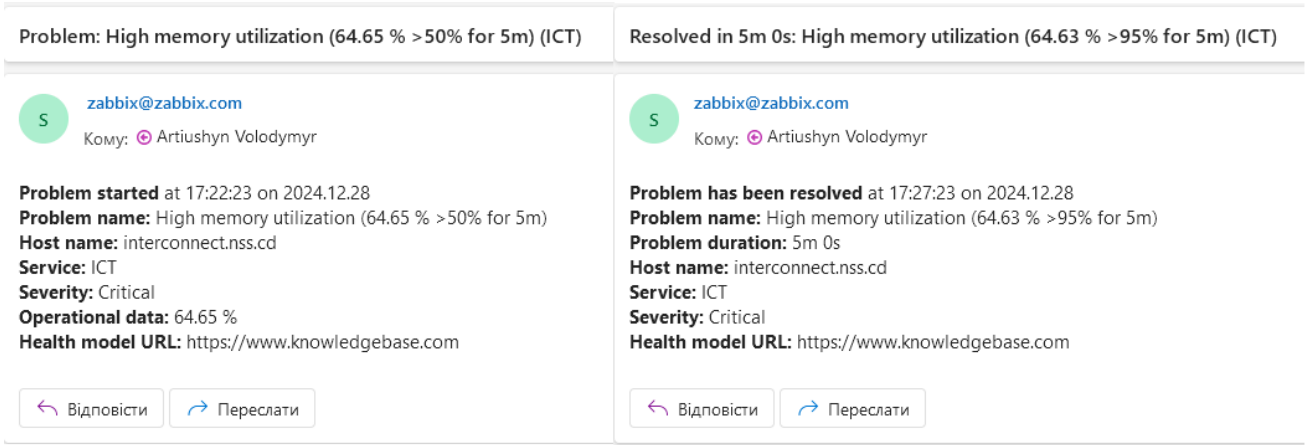


Рис. 3.10 Отримані попередження на електронну пошту з приводу початку та закінчення критичної проблеми

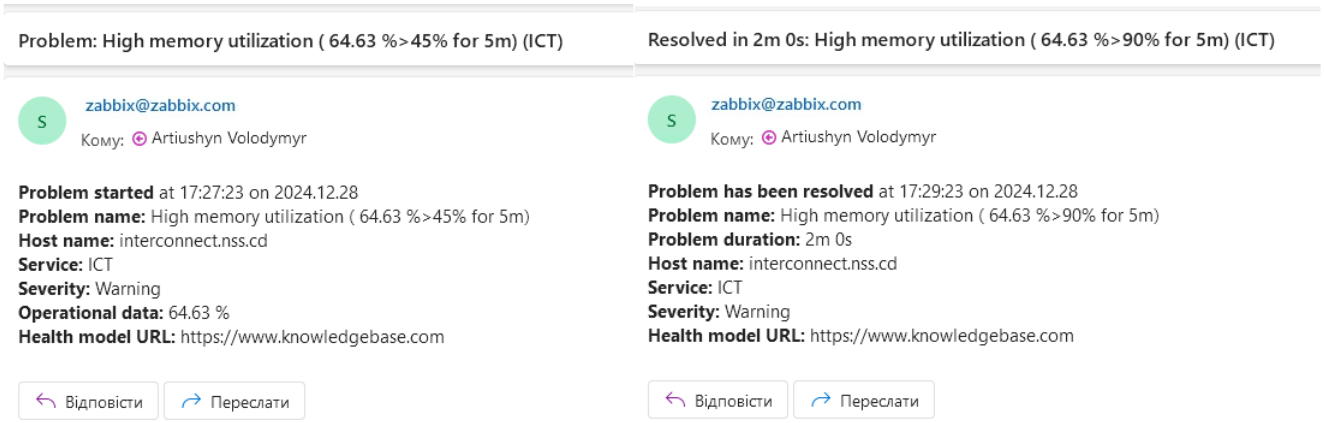


Рис. 3.11 Отримані попередження на електронну пошту з приводу початку та закінчення проблеми з важливістю “Попередження”

Аналогічно, тестуємо решту метрик з Табл. 2.1, встановлюючи порогові значення нижче ниж поточні значення метрик (Рис.3.12). Для порта “ens192” імітуємо аварію примусово переводячи його в непрацюючий режим на обладнанні серверу. Впевнюємося, що на електронну пошту приходять листи у момент початку та закінчення проблем.

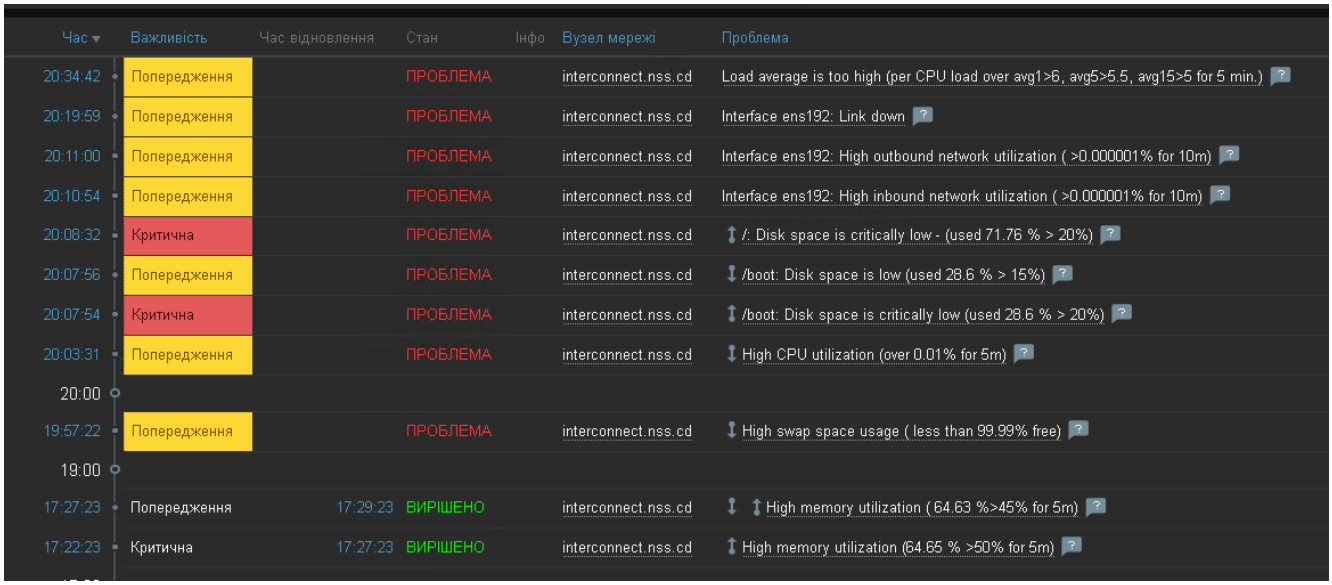


Рис. 3.12 Відображення проблем на вкладці “Проблеми” для метрик з Табл. 2.1

Моніторинг утилізації БД Oracle, який був описаний у підрозділі 2.2, будемо тестувати, також, задаючи необхідні порогові значення. Порогові значення раніше були встановлені макросами на 95% та 98%.

Аналізуючи останні дані, бачимо, що більше половини таблиць має утилізацію більше 30%. Встановимо поріг для попередження на 30% та для критичної аварії на 40% та отримаємо результати (Рис. 3.13).

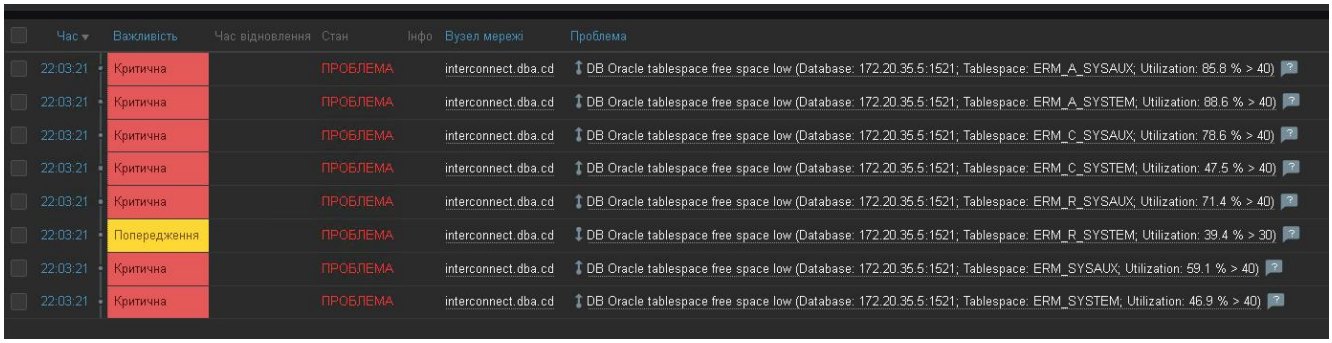


Рис. 3.13 Відображення проблем на вкладці “Проблеми” для метрик з моніторингу утилізації БД Oracle

Метрику моніторингу доступності з'єднання з БД (елемент даних “SQL connection status”), яка перевіряє відсутність доступу запитом “select count(*) from dual”, тестуємо разом з адміністратором БД. Для цього користувачеві БД zabbix адміністратор тимчасово блокує можливість виконання будь яких SQL команд. У відповідь скрипт (Рис. 2.25) повертає помилку і спрацьовує тригер. Перевіряємо електронну пошту та фіксуємо інформаційні листи, аналогічні листам на Рис. 3.10 та Рис. 3.11, які нас інформують про проблеми з БД Oracle.

Моніторинг комутатора, який був описаний у підрозділі 2.3, будемо тестувати, також, задаючи необхідні порогові значення. У нашому випадку у комутатора, як було показано на Рис 3.5 не працює 2 порти. Тому вже зараз маємо аварію по цим 2-м портам. Об’єм трафіку, який проходить через спостережувані інтерфейси досить незначний. Тому, для примусового спрацювання тригерів встановлюємо процент утилізації інтерфейсу 5% замість 80%.

Як бачимо, на Рис. 3.14, тригери спрацювали на завантаження інтерфейсів по вхідному та вихідному трафіку за умови перевищення утилізації більше 5%. Перевіряємо електронну пошту та фіксуємо інформаційні листи, аналогічні листам на Рис. 3.10 та Рис. 3.11, але вже по завантаженню інтерфейсів.

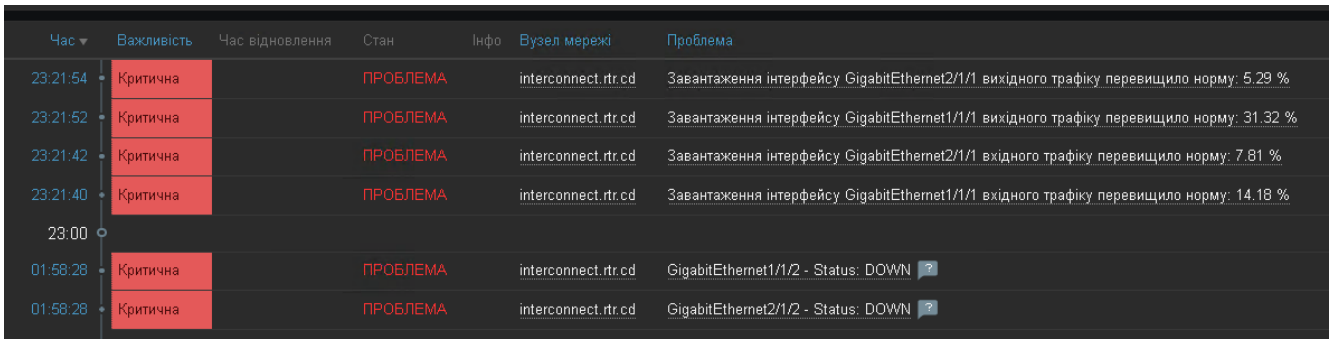


Рис. 3.14 Відображення проблем на вкладці “Проблеми” для метрик з моніторингу комутатора

Таким чином, було протестовано роботу розробленої інформаційної системи моніторингу мобільного оператора та коректне спрацювання автоінформування по відхиленню ключових показників від нормтивних значень.

Далі, як показано на Рис. 2.45 останнім етапом є документування результатів, яке включає:

- детальний опис метрик;
- опис порогів або умов спрацювання тригерів;
- інструкції для чергового персоналу при винекненні аварійних ситуацій.

Інструкції для чергового персоналу включають дії, які треба виконати при виникненні той чи іншої аварійної ситуації (переслати листа, зателефонувати та інше). Дії, у свою чергу, можуть поділятися на дії, які виконуються у робочі години, або дії, які виконуються у неробочі години.

Конкретний опис системи та документування результатів залежить від специфіки роботи системи, яка знаходиться на моніторингу, внутрішніх стандартів, процедур та політик у компанії оператора.

ВИСНОВКИ

В результаті виконання даної кваліфікаційної роботи була розроблена інформаційна системи моніторингу мобільного оператора на базі системи Zabbix.

Для цього було проаналізовано основні проблеми сучасних мобільних мереж, типи обладнання, які використовуються мобільними операторами, сучасні системи моніторингу мобільного оператора та особливості системи Zabbix, як найбільш перспективної системи моніторингу для мереж мобільних операторів.

Щоб найбільш повно розкрити можливості системи Zabbix, була створена та встановлена на моніторинг модель системи білінгу міжоператорського трафіку, яка складається з трьох різних мережевих елементів з різними завданнями, протоколами та портами моніторингу: сервер з ОС Linux, сервер з встановленою БД Oracle та комутатор.

У ході практичного виконання даної кваліфікаційної роботи були виконани наступні завдання дослідження:

1. Проаналізовані вхідні дані та вибрані метрики, які треба додати на моніторинг для різних типів обладнання, яке входить у систему встановлену на моніторинг;
2. Розроблена інформаційна система моніторингу мобільного оператора на базі системи Zabbix;
3. Виконана візуалізація результатів роботи у вигляді інформаційної панелі, яка дає змогу оператору візуально бачити основні показники системи білінгу міжоператорського трафіку та вчасно виявляти аномальні відхилення значень метрик або відсутність даних;
4. Виконано дослідження роботи розробленої системи моніторингу та отримані результати тестування;
5. Налаштоване автоматичне інформуваннння про відхилення основних метрик від нормальних значень на електронну пошту.

В ході виконання проекту, Zabbix показав себе як гнучка та потужна система моніторингу з відкритим кодом, яка забезпечує повний контроль над

різноманітним обладнанням мобільного оператора з різними протоколами доступу.

Теоретична, методична та практична значущість отриманих результатів полягає в можливості застосування розробленої інформаційної системи моніторингу для удосконалення експлуатаційної діяльності та підвищення якості обслуговування мережевих елементів у середовищі мобільного оператора або провайдера.

Розроблену систему можна використовувати як базовий шаблон для встановлення на моніторинг інформаційних систем мобільного оператора які містять у своєму складі сервери на базі ОС Linux, комутатори або БД Oracle.

ПЕРЕЛІК ПОСИЛАНЬ

1. Андреа Далле Вакке Zabbix. Практичний посібник 2-е вид., М.: ДМК Пресс, 2017. – 356 с.
2. 4G. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/4G.
3. 5G. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/5G.
4. SIM-картка. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/SIM-%D0%BA%D0%B0%D1%80%D1%82%D0%BA%D0%B0.
5. Operations support system. [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Operations_support_system
6. Network Management System (NMS). [Електронний ресурс] – Режим доступу до ресурсу: https://www.omgwiki.org/dido/doku.php?id=dido:public:ra:xapend:xapend.a_glossary:n:nms
7. Керування мережею. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D1%8E
8. Zabbix. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Zabbix
9. Посібник Zabbix. [Електронний ресурс] – Режим доступу до ресурсу: https://www.zabbix.com/documentation/6.4/ua/manual
10. SNMP. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/SNMP
11. Download and install Zabbix. [Електронний ресурс] – Режим доступу до ресурсу: https://www.zabbix.com/download_agents

12. Low-level discovery. [Електронний ресурс] – Режим доступу до ресурсу: https://www.zabbix.com/documentation/6.4/ua/manual/discovery/low_level_discovery

13. Список номерів портів TCP та UDP. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D0%BD%D0%BE%D0%BC%D0%B5%D1%80%D1%96%D0%B2_%D0%BF%D0%BE%D1%80%D1%82%D1%96%D0%B2_TCP_%D1%82%D0%B0_UDP

14. Tablespace. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Tablespace

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА
на тему: «Розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix»

на здобуття освітнього ступеня магістра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

Виконав: здобувач вищої освіти, група ІСДМ-64
Володимир АРТЮШИН
Керівник: доктор технічних наук, професор
Вікторія ЖЕБКА

Київ-2024

Актуальність теми. Мобільна мережа оператора за останні два десятиліття перетворилася у велику та складну систему з різними типами обладнання та багаторівневою архітектурою. Навіть вихід з ладу одного мережевого елементу або каналу зв'язку між елементами може вплинути на працездатність сервісів або мережі в цілому. Саме тому побудова ефективної системи моніторингу є одним з пріоритетних завдань для операторів, що дає змогу швидко виявити аварійні ситуації та мінімізувати їх наслідки.

Мета і завдання дослідження є розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix.

Завдання дослідження:

1. Проаналізувати вхідні дані та вибрати метрики, які треба встановити на моніторинг;
2. Розробити інформаційну систему моніторингу мобільного оператора на базі системи Zabbix;
3. Дослідити роботу розробленої системи моніторингу та отримати результати тестування.

Об'єктом дослідження є процес розробки інформаційної системи моніторингу мобільного оператора та встановлення на моніторинг хостів системи.

Предметом дослідження є система моніторингу Zabbix версії 6.4, сервери з встановленою ОС Linux та мережевий комутатор.

Методи дослідження. У процесі роботи над проєктом були проаналізовані основні можливості системи моніторингу Zabbix, розроблена інформаційна система моніторингу мобільного оператора та продемонстровані отримані результати.

Основним завданням кваліфікаційної роботи є розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix.

Дистрибутив системи моніторингу Zabbix версії 6.4 встановлено на сервер під керуванням операційної системи Linux. Керування на налагодження моніторингу всіх мережевих елементів відбувається через WEB-інтерфейс.

Для прикладу розглядається система білінгу міжоператорського трафіку, яка складається з двох серверів на базі ОС Linux (interconnect.nss.cd та interconnect.dba.cd), на одному з яких встановлена база даних (БД) Oracle. Додатково, по протоколу SNMP моніториться мережевий комутатор interconnect.rtr.cd.

Сервер interconnect.nss.cd
(ОС Linux)

Сервер interconnect.dba.cd
(ОС Linux + БД Oracle)

Комутатор interconnect.rtr.cd

Основні метрики для моніторингу серверів з ОС Linux

Таблиця 1

Назва метрики	Опис технічного параметру	Одиниця виміру	Пороги та критичність
Memory utilization in %	Утилізація пам'яті	%	Попередження: > 90% за 5 хв.
			Критична: > 95% за 5 хв.
SWAP utilization in %	Утилізація файлу підкачки (SWAP) в %	%	Попередження: > 80% за 5 хв.
CPU total utilization in %	Утилізація процесору	%	Попередження: > 90% за 5 хв.
CPU load avarage in %	Середне навантаженя на процесор (5, 15 хв. інтервал)	число	Попередження: > 1.5
Space utilization in %	Використання простору в % (для кожного розділу файлової системи)	%	Попередження: util. > 85%
			Критична: util. > 95%
High inbound network utilization in %	Завантаження вхідного трафіку порівняно з максимальною пропускною здатністю мережевого інтерфейсу (для кожного мережевого інтерфейсу).	%	Попередження: > 90% за 10 хв.
High outbound network utilization in %	Завантаження вихідного трафіку порівняно з максимальною пропускною здатністю мережевого інтерфейсу (для кожного мережевого інтерфейсу).	%	Попередження: > 90% за 10 хв.
High error rate	Високий рівень помилок (для кожного мережевого інтерфейсу).	число	Попередження: > 5 за 5 хв.
Link down	Мережеве з'єднання не працює (для кожного мережевого інтерфейсу).	число	Попередження: = down

Моніторинг баз даних Oracle

Таблиця 2

Назва метрики	Опис технічного параметру	Одиниця виміру	Пороги та критичність
Tablespace utilization in %	Утилізація таблиць БД Oracle	%	Попередження: > 95%
			Критична: > 98%
SQL connection status	Моніторинг доступності з'єднання до БД	число	Критична: немає з'єднання з БД

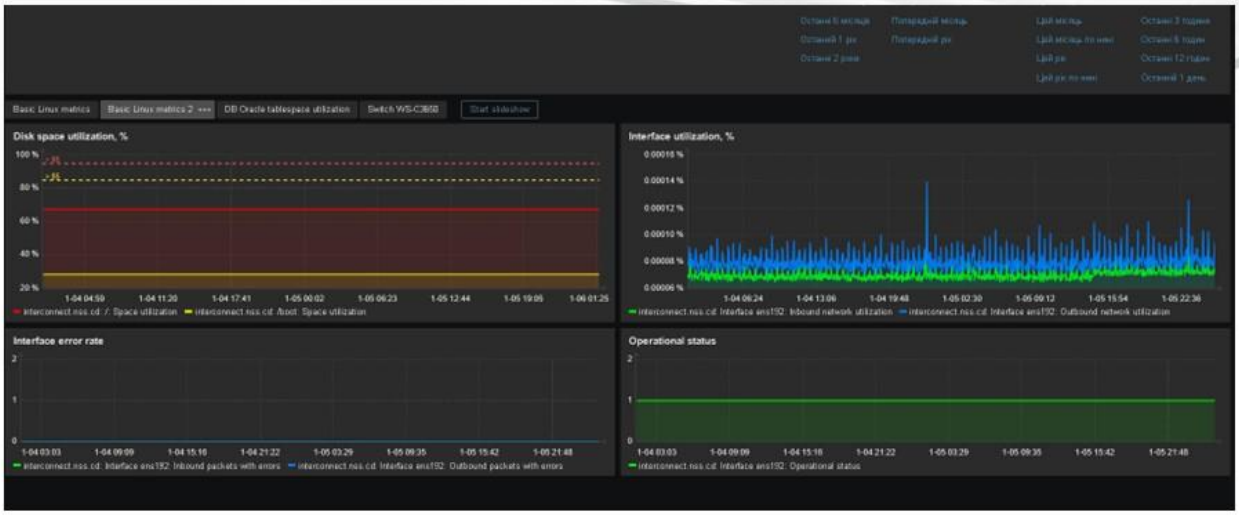
Основні метрики для моніторингу комутатору по протоколу SNMP

Таблиця 3

Назва метрики	Опис технічного параметру	Одиниця виміру	Пороги та критичність
Стан портів комутатора	Виявлення портів комутатора та їх робочого стану	число	Критична: = 2 (порт не працює)
Обсяг вхідного трафіку на портах	Завантаження вхідного трафіку порівняно з максимальною пропускною здатністю порту (для кожного порту).	%	Критична: > 80 %
Обсяг вихідного трафіку на портах	Завантаження вихідного трафіку порівняно з максимальною пропускною здатністю порту (для кожного порту).	%	Критична: > 80 %

Для відображення метрик та інших графічних результатів розробником виконується візуалізація у вигляді дашбордів або інформаційних панелей. Було вирішено зробити інформаційну панель з чотирма вкладками. Для полегшення візуалізації (треба щоб панелі були видимі на одному екрані без необхідності прокрутки) дві перші вкладки будуть відображати основні метрики ОС Linux з Табл. 1.

Вкладка №1 з основними метриками ОС Linux



Вкладка №2 з основними метриками ОС Linux

Метрики утилізації БД Oracle та моніторингу доступності з'єднання з БД (Вкладка №3)

Метрики трафіку та стану портів комутатору (Вкладка №4)

У ході практичного виконання даної кваліфікаційної роботи були виконани наступні завдання дослідження:

1. Проаналізовані вхідні дані та вибрані метрики, які треба додати на моніторинг для різних типів обладнання, яке входить у систему встановлену на моніторинг;
2. Розроблена інформаційна система моніторингу мобільного оператора на базі системи Zabbix;
3. Виконана візуалізація результатів роботи у вигляді інформаційної панелі, яка дає змогу оператору візуально бачити основні показники системи білінгу міжоператорського трафіку та вчасно виявляти аномальні відхилення значень метрик або відсутність даних;
4. Виконано дослідження роботи розробленої системи моніторингу та отримані результати тестування;
5. Налаштоване автоматичне інформуваннння про відхилення основних метрик від нормальних значень на електронну пошту.

Висновки

В результаті виконання даної кваліфікаційної роботи була розроблена інформаційна системи моніторингу мобільного оператора на базі системи Zabbix.

Для цього було проаналізовано основні проблеми сучасних мобільних мереж, типи обладнання, які використовуються мобільними операторами, сучасні системи моніторингу мобільного оператора та особливості системи Zabbix, як найбільш перспективної системи моніторингу для мереж мобільних операторів.

Щоб найбільш повно розкрити можливості системи Zabbix, була створена та встановлена на моніторинг модель системи білінгу міжоператорського трафіку, яка складається з трьох різних мережевих елементів з різними завданнями, протоколами та портами моніторингу: сервер з ОС Linux, сервер з встановленою БД Oracle та комутатор.

В ході виконання проекту, Zabbix показав себе як гнучка та потужна система моніторингу з відкритим кодом, яка забезпечує повний контроль над різноманітним обладнанням мобільного оператора з різними протоколами доступу.

Теоретична, методична та практична значущість отриманих результатів полягає в можливості застосування розробленої інформаційної системи моніторингу для удосконалення експлуатаційної діяльності та підвищення якості обслуговування мережевих елементів у середовищі мобільного оператора або провайдера.

Розроблену систему можна використовувати як базовий шаблон для встановлення на моніторинг інформаційних систем мобільного оператора які містять у своєму складі сервери на базі ОС Linux, комутатори або БД Oracle.

Апробація результатів та публікації.

Артюшин В.В. "Розробка інформаційної системи моніторингу мобільного оператора на базі системи Zabbix". Тези доповіді на II всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу». – Київ, 18 листопада 2024 р.

Артюшин В.В. "Система моніторингу Zabbix як універсальний інструмент контролю обладнання дата-центрів". Тези доповіді та II міжнародній науково-практичній конференції «Сучасні аспекти діджиталізації та інформатизації в програмній та комп'ютерній інженерії» – Київ, 19-21 грудня 2024 р.

ДЯКУЮ ЗА УВАГУ!