Лабораторна робота №16

Дашборди та тригери з Telegram-сповіщеннями в Zabbix.

Мета: набути практичних навичок створення користувацького дашборду в Zabbix, налаштування SNMP-тригерів для визначення критичних подій та інтеграції системи моніторингу з Telegram для автоматичної відправки сповіщень про стан хостів.

Інструменти: гіпервізор VirtualBox, модель комп'ютерної мережі.

Теоретичні відомості

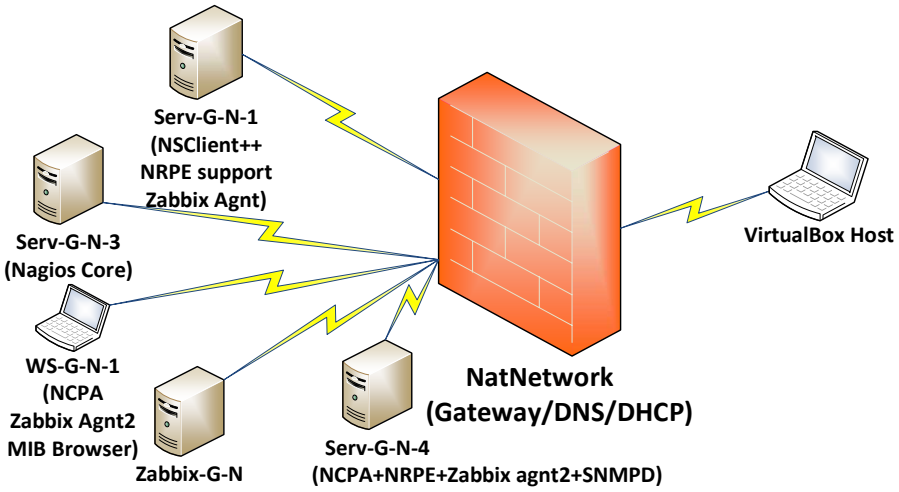


Рис. 16.1. Топологія мережі

На рис. 16.1 наведена модель комп'ютерної мережі, побудована під час виконання попередніх лабораторних робіт. На сервері Serv-G-N-3 розгорнуто систему моніторингу на базі Nagios 4.X. На сервері Zabbix-G-N працює сервер Zabbix з базовими налаштуваннями. В обох моніторингових системах налаштоване спостереження за Serv-G-N-1, WS-G-N-1, Serv-G-N-4. На хосту Serv-G-N-4 налаштований сервіс SNMP-серверу. У попередній лабораторній роботі на сервері Nagios 4.X (Serv-G-N-3) налаштовано моніторинг для збору даних через протокол SNMP.

У попередній лабораторній роботі був створений користувацький графік (рис.16.2).

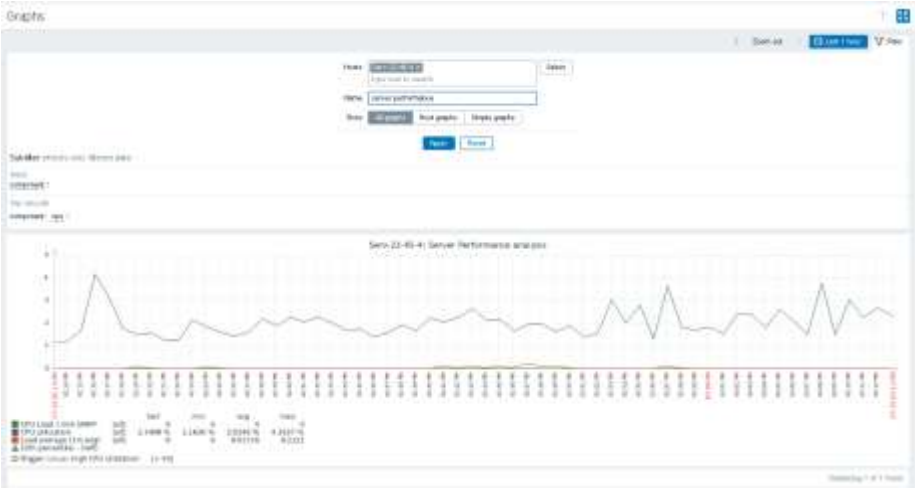


Рис. 16.2. Графіку взаємозалежності використання ЦП через SNMP та стандартні метрики та середнього використання ЦП для хосту Serv-22-45-4

Створення класичного Dashboard

У Zabbix дашборди використовуються для швидкого огляду стану інфраструктури, а також для аналізу її продуктивності. Основні типи корисних дашбордів: оглядовий дашборд (Overview Dashboard), мережевий дашборд, що аналізує стан мережевих пристроїв та інтерфейсів, дашборд для моніторингу сервісів, що слідкує

за доступністю та продуктивністю окремих сервісів, безпековий дашборд, що відстежує інциденти безпеки та спроб проникнення та інші.

Створимо новий користувацький DashBoard на основі побудованого нами графіку. До нього включимо графік Server Performance Analysis, відображення логу проблем хосту, годинник та географічну мапу. Переходимо у меню [DashBoards] та обираємо у меню кнопки [Actions] пункт [Create New].

Власника залишаємо без змін, назва Serv-G-N-4 BSNM-N, де N-номер варіанту. Період оновлення залишаємо 30 сек, або змінюємо на 1 хв. Після цього у робочій області створюваного DashBoard з'являється фрейм першого елементу. Налаштовуємо його тип як Graph (classic) з назвою серверу та обираємо у якості графіку створений нами на попередніх кроках графік [Server Performance Analysis].

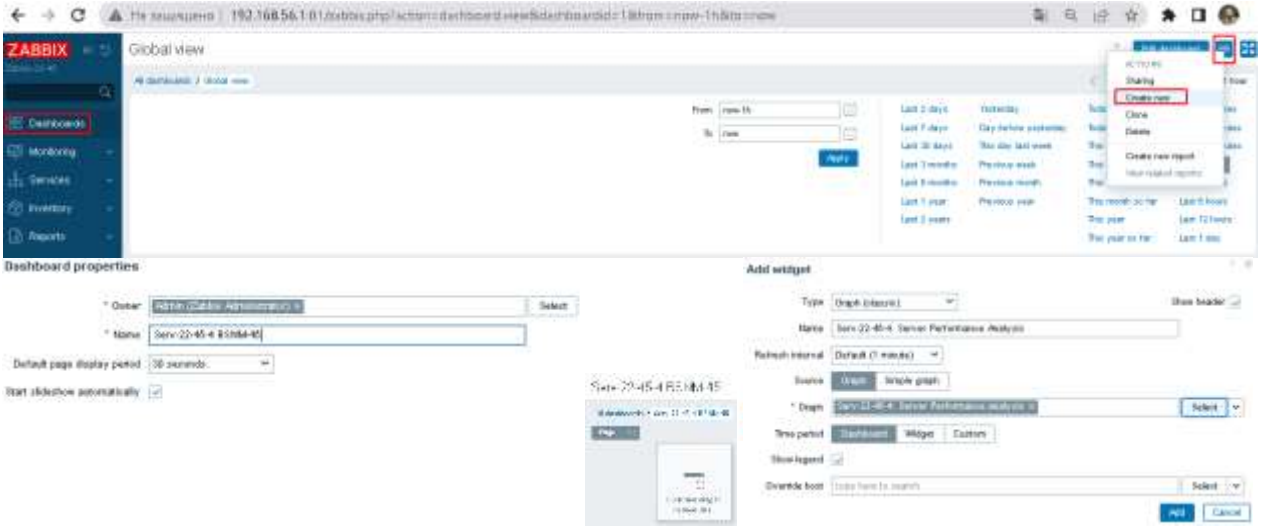


Рис. 16.3. Створення DashBoard "Serv-22-45-4 BSNM-45" та налаштування першого елементу – графіку.

У якості другого елементу DashBoard налаштовуємо «віджет проблем» хосту. Третій елемент – віджет годинника. Четвертий елемент – географічна мапа, де у полі "Initial View" задано координати м. Житомир ☺. Якщо Ви розміщаєте свій сервер Serv-G-N-3 у Бангкоку, введіть його координати в цьому полі ☺.

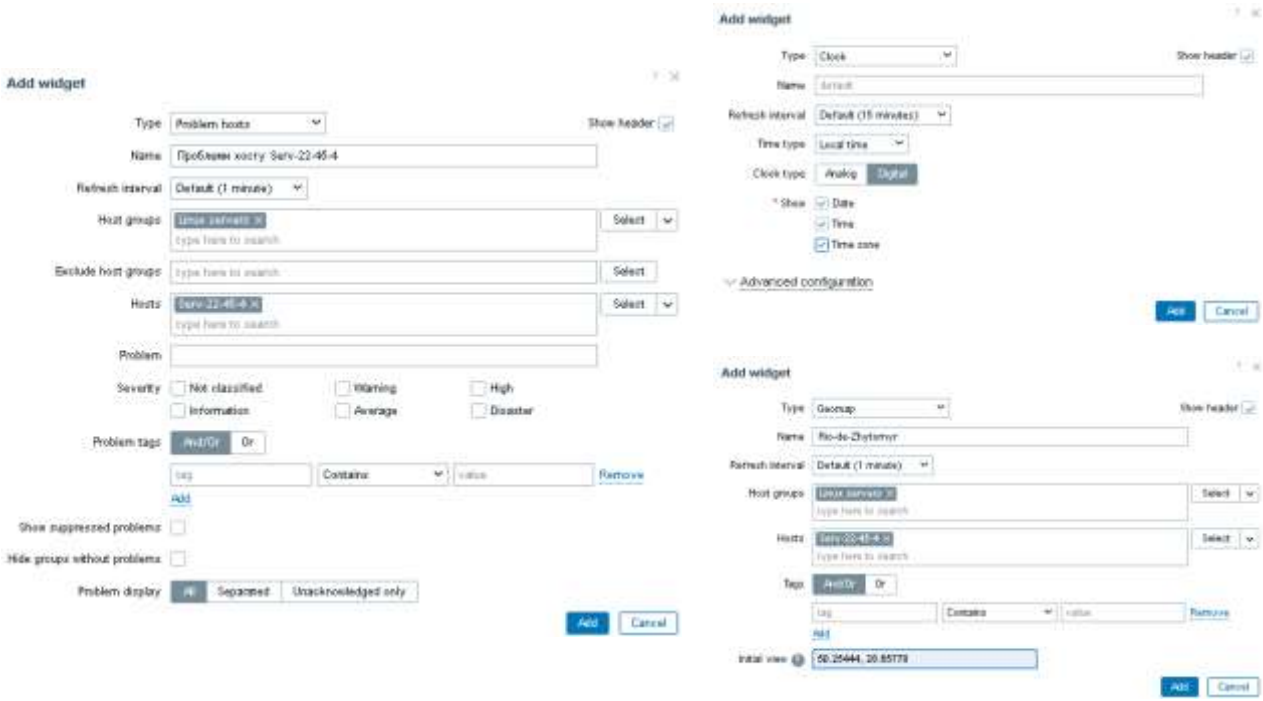


Рис. 16.4. Додавання до DashBoard "Serv-22-45-4 BSNM-45" наступних елементів.

Виконані дії дозволяють створити простий, але наочний DashBoard"Serv-G-N-4 BSNM-N", що складається з чотирьох елементів: користувацького графіка, списку проблем хосту, годинника та географічної мапи.

Слід зазначити, що під час створення DashBoard кількість елементів може бути значно більшою, а їхнє розміщення може охоплювати кілька сторінок для зручності моніторингу та аналітики.

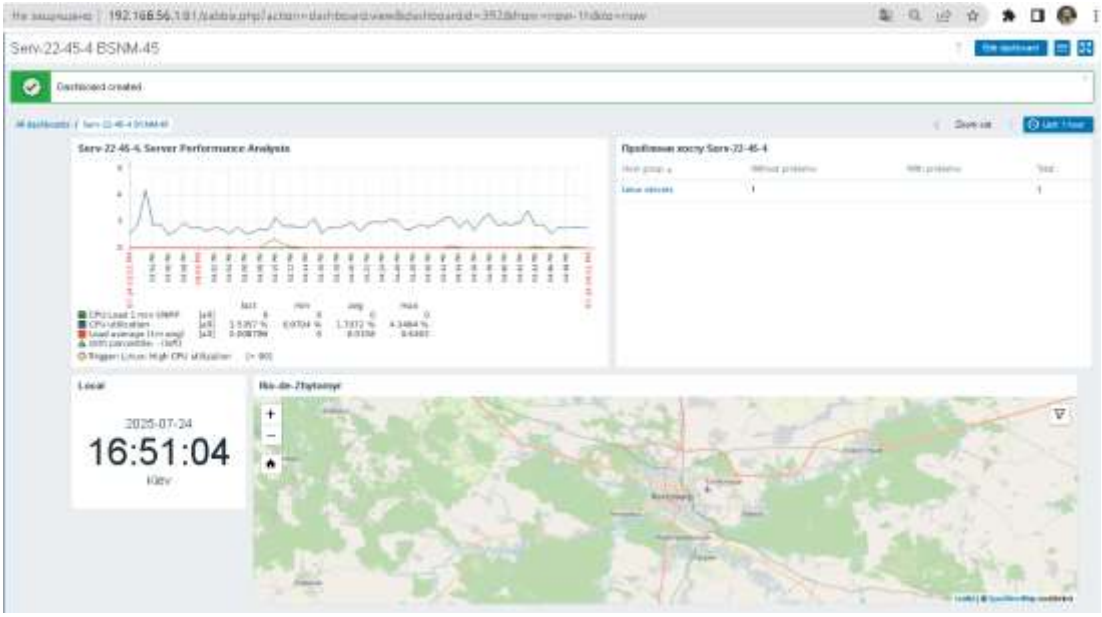


Рис. 16.5. Загальний вигляд створеного DashBoard “Serv-22-45-4 BSNM-45”.

Налаштування тригерів

Тригери (Triggers) — це механізм у Zabbix, який використовується для автоматичного аналізу даних, що надходять від елементів (Items), і визначення, чи є стан моніторингу нормальним або проблемним. Тригер спрацьовує на основі заданого логічного виразу (expression), який оцінює отримані дані та порівнює їх із певними пороговими значеннями.

Тригери використовуються для виявлення проблем - визначення аномалій, наприклад, завантаження CPU понад 80%, низький вільний простір на диску, або відсутність даних від пристрою. З тригерами пов'язані дії (Actions), які можуть надсилати повідомлення (email, Telegram, Slack) або виконувати команди (наприклад, перезапуск служби). Тригери мають рівні важливості (severity), що дозволяє класифікувати проблеми за критичністю (наприклад, від «Not classified» до «Disaster»). Тригери сприяють автоматичному реагуванню на інциденти без потреби постійного ручного втручання.

Алгоритм роботи тригера наступний - дані отримуються від Item: Zabbix збирає дані через SNMP, агентів, скрипти тощо. Вираз тригера аналізує дані: наприклад, перевіряє, чи значення CPU > 80%. Якщо вираз оцінюється як true, тригер змінює стан на Problem. Коли умова більше не виконується, тригер повертається до стану OK. Для створення цього тригеру переходимо у меню [Data Collection] – [Hosts] та обираємо меню [Triggers] відповідного хосту. У нашому випадку це буде [Serv-22-45-4] – [Triggers(42)]. У правому верхгьому кутку завантаженого вікна [Triggers] натискаємо кнопку [Create Trigger] . Вантажиться дочірнє вікно [New trigger], де пишемо назву тригеру «High CPU Usage» та натискаємо Add для написання вмісту поля Expression. По кнопці [Select] обираємо створений нами раніше Item “CPU Load 1 min SNMP” Function – обираємо функцію середнього значенняза період (avg), у якосіт періоду обираємо 5 періодів (по 1 хв) і у виразі Result ставимо умову >80.

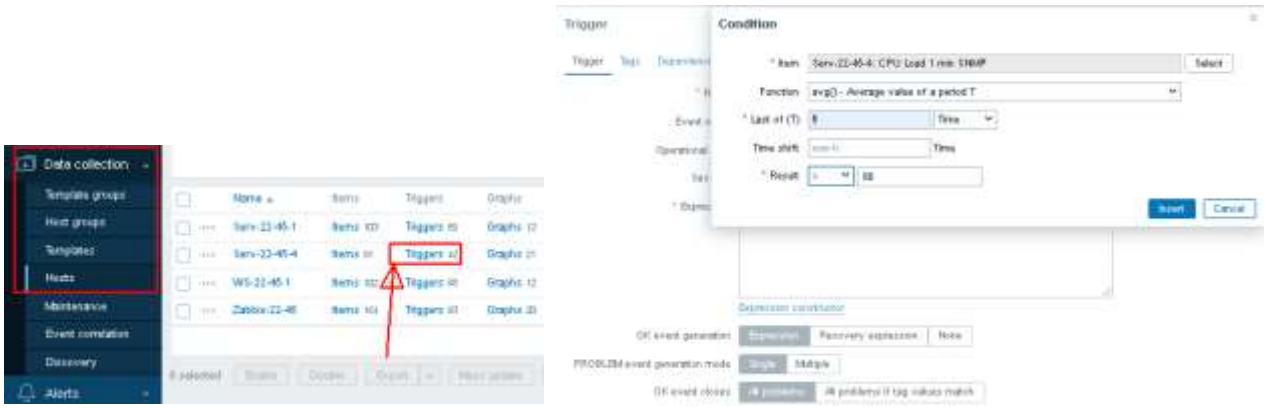


Рис. 16.6. Створення Triger «High CPU Usage» для Serv-22-45-4.

Натискаємо кнопку [Insert] вікна [Condition] і знову потрапляємо до вікна [New trigger]. Ще раз переглядаємо отримані заповнення, та обираємо значення [Severity] High.

Отриманий вираз

avg(/Serv-22-45-4/cpu.load.1min.snmp,5)>80

перевіряє, чи середнє значення метрики cpu.load.1min протягом п'яти перевірок перевищує 80. Якщо необхідно додати резервний вираз для ситуації, коли середнє завантаження CPU на інших інтервалах часу також має перевищення, тригер може бути розширений з використанням операторів OR чи AND. Наприклад:

avg(/Serv-22-45-4/cpu.load.1min.snmp,5)>80 or avg(/Serv-22-45-4/cpu.load.1min.snmp,3)>95

У цій умові достатньо щоб хоча б один вираз виконався (>80 за 5 перевірок або >95 за 3):

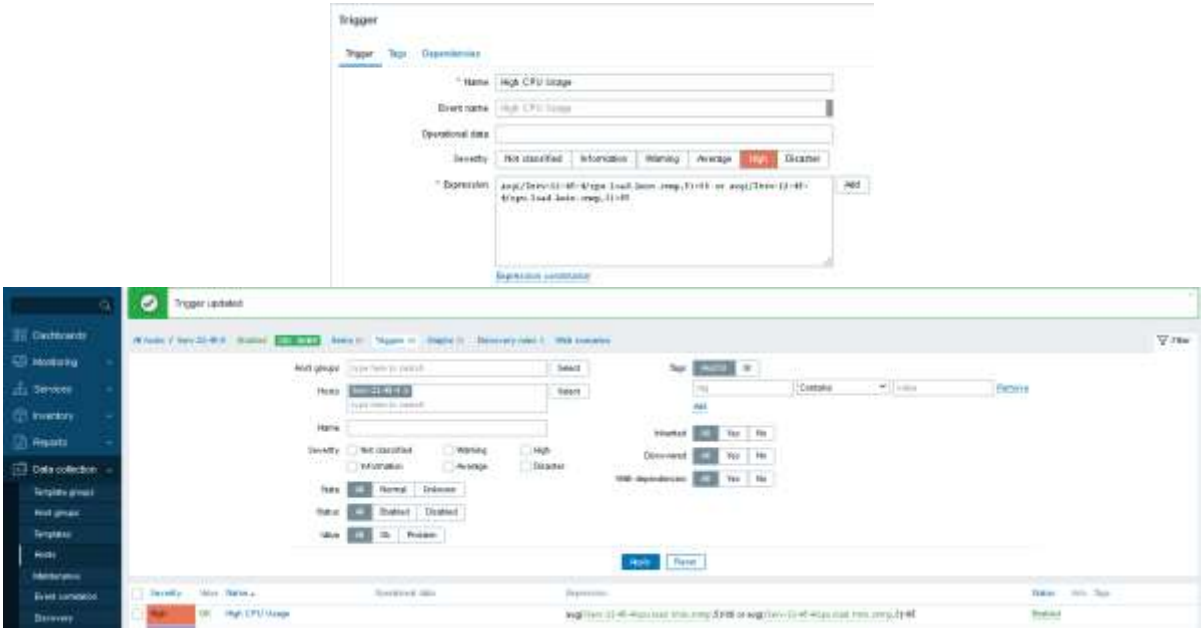


Рис. 16.7. Створення Triger «High CPU Usage» для Serv-22-45-4.

Налаштування сповіщень

У поточній на момент написання цього документу версії Zabbix присутні вбудовані шаблони для всіх основних медіа-типів сповіщень. Переглянути їх можливо у меню [Alerts] – [Media Types].

Налаштуємо сповіщення для створеного на попередньому кроці тригеру «High CPU Usage» для серверу Serv-22-45-4 за допомогою Media Type Telegram. Для цього, створимо користувача Zabbix, від імені якого буде виконуватись ця дія. Переходимо у меню [Users] – [Users] і у правому верхньому кутку вікна натискаємо кнопку [Create User]. Ім'я користувача відповідає шаблону User-G-N. Йому присвоєна роль звичайного користувача (User role) та він є членом групи Internal.

У меню [Users] – [User Roles] можна переглянути, відредагувати чи створити ролі користувачів.

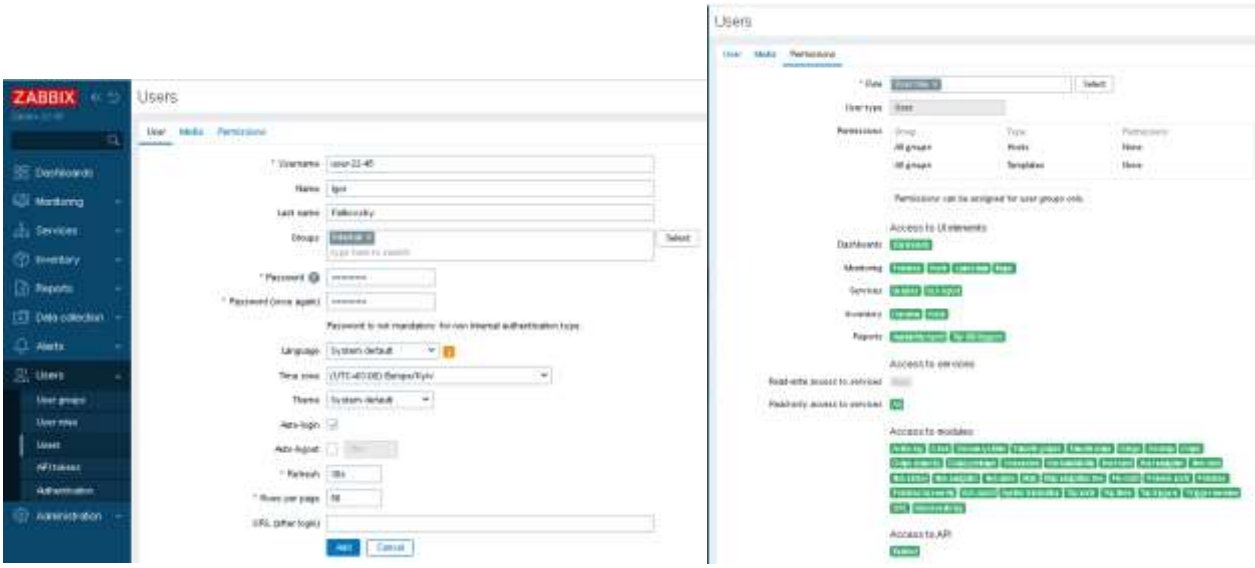


Рис. 16.8. Створення користувача user-22-45.

Переходимо у меню [Alerts] – [Media Types] тип [Telegram] та активуємо натисканням його статусу у пункті. Статус має змінитися з [Disabled] на [Enabled]. Переходимо до редагування налаштувань цього Media.

Рис. 16.9. Активація media для Telegram.

У додатку 16.1 описано створення та виділення Telegram bot та отримання (генерація) необхідних для налаштування сповіщень у телеграм token та id створеного бота.

Натискаємо меню [Alerts] – [Media Types] та у діалозі налаштувань для Telegram заповнюємо поля api_chat_id та api_token:

Поле	Значення	Налаштування
api_chat_id	{ALERT.SENDTO}	отримувач повідомлення (**сюди передається ID чату**).
api_token		**Token Telegram bot**

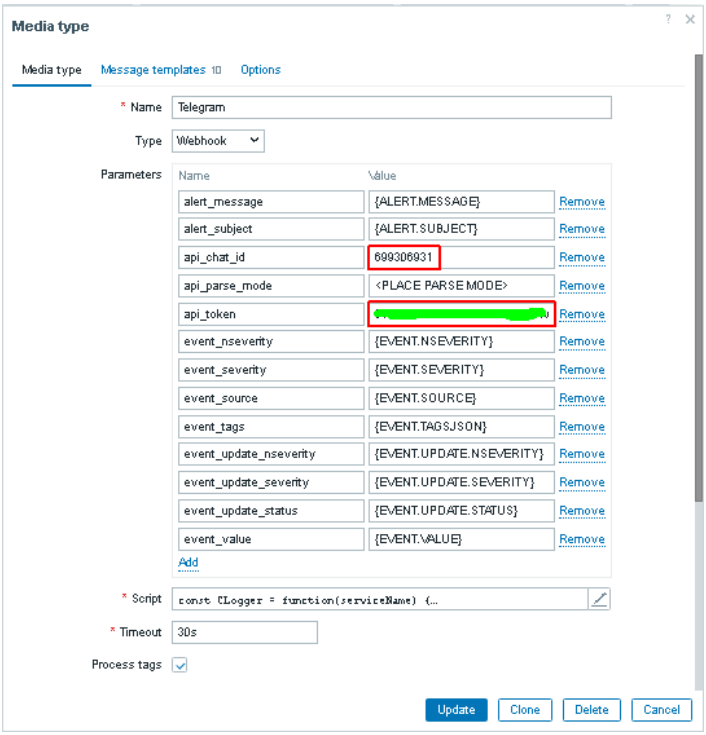
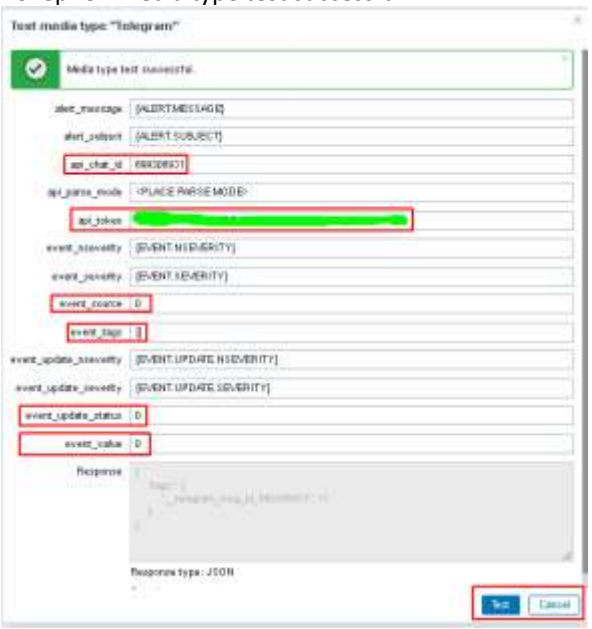


Рис. 16.10. Заповнення Media Type для Telegram.

Якщо всі поля вірно заповнено, виклик вікна Test через відповідне меню Media Type для Telegram поверне «Media type test successful.»



Зверніть увагу, що для проходження тесту у Media Type Telegram на час тесту потрібно присвоїти полям ***event_source***, ***event_update_status*** та ***event_value*** значення від 0 до 2, а полю ***event_tags*** значення [] (порожній JSON-масив). Ці налаштування не зберігаємо, а після тесту натискаємо кнопку Cancel.

Результат успішного тестування наведено нижче.

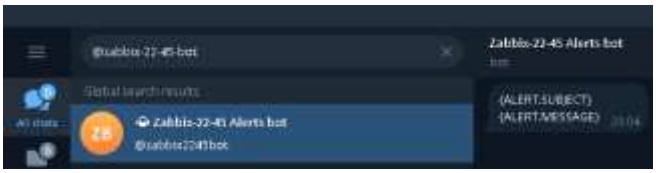


Рис. 16.11. Тестування Media Type для Telegram.

Додаємо медіа-тип Telegram до створеного на попередніх кроках користувача User-G-N. Переходимо у меню [Users] – [Users] та обираємо користувача (user_22_45). Вкладка [Media] – [Add]. Заповнюємо

Type: Telegram Send to: 699306931

When active: 1-7,00:00-24:00

Use if severity: обираємо всі рівні або лише потрібні для цього користувача.

Enabled: Так

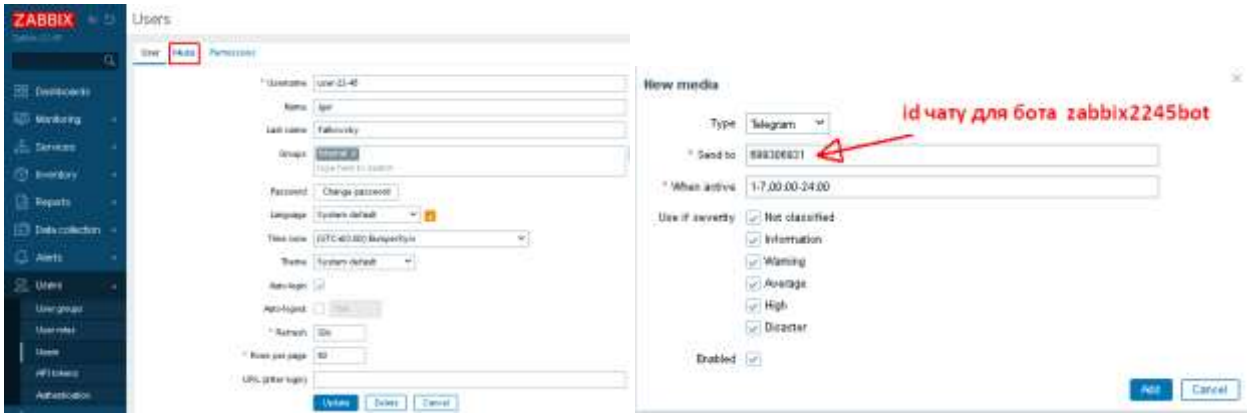


Рис. 16.12. Налаштування Media для користувача User-22-45 у Zabbix

Налаштовуємо дію (Action) для сповіщень у Telegram. Переходимо у меню [Allert] – [Actions] – [Trigger Actions] та обираємо у верхньому правому кутку вікна [Trigger Actions] кнопку [Create Action]. На рис. 16.23 показаний приклад створення сповіщення для окремої події на сервері Serv-22-40-1

Якщо все налаштовано правильно, має прийти сповіщення в Telegram. ✉

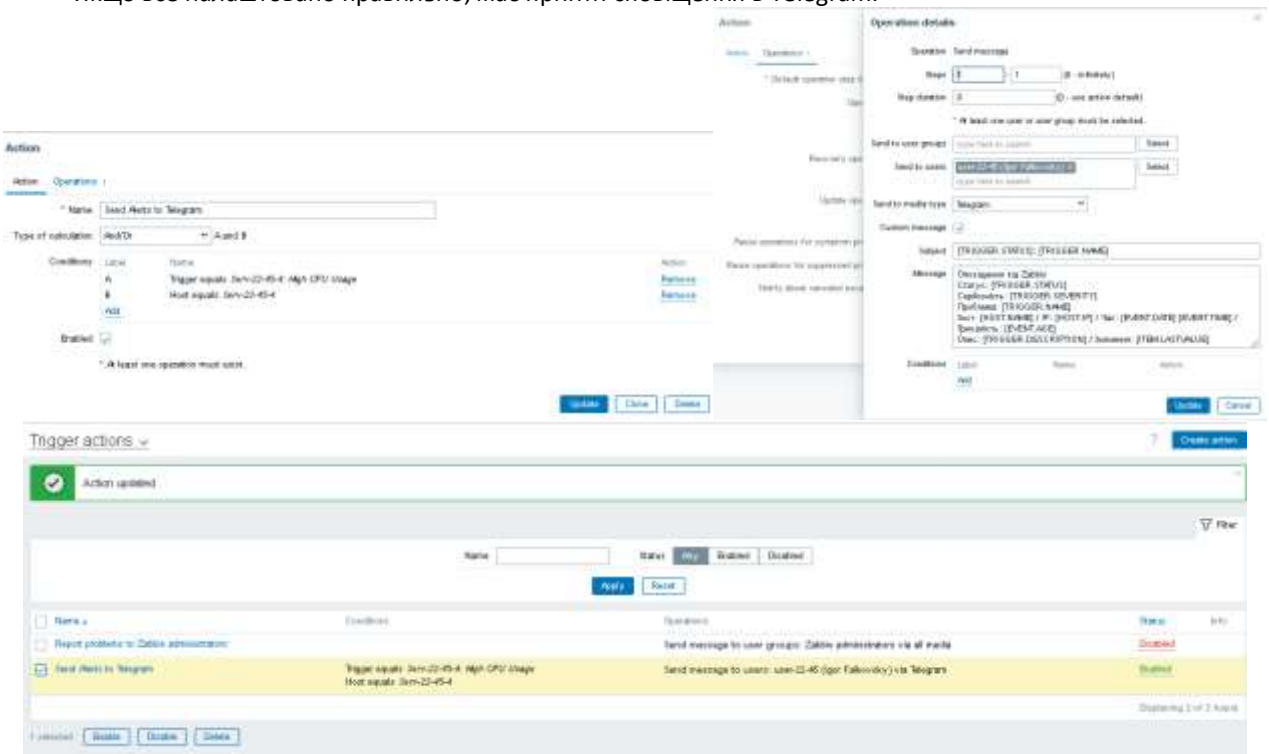


Рис. 16.13. Налаштування Trigger Actions на відправку сповіщення у Telegram для події "CPU interrupt time is too high" для серверу Serv-22-45-4

Ось готовий текст для поля Message (закладка Operation details) для Telegram у Zabbix, який містить усю потрібну інформацію про подію:

Оповіщення від Zabbix
Статус: {TRIGGER.STATUS}
Серйозність: {TRIGGER.SEVERITY}
Проблема: {TRIGGER.NAME}
Хост: {HOST.NAME} / IP: {HOST.IP} / Час: {EVENT.DATE} {EVENT.TIME} / Тривалість: {EVENT.AGE}
Опис: {TRIGGER.DESCRIPTION} / Значення: {ITEM.LASTVALUE}

Завдання до лабораторної роботи

1. У веб-інтерфейсі Zabbix створіть новий дашборд із назвою "Serv-G-N-4 BSNM-N" (де N – номер вашого варіанту). та періодом оновлення 1 хвилина. Дашборд має містити чотири елементи Graph classic - [Server Performance Analysis], список проблем, годинник, географічну мапу з початковими координатами м. Житомир (або координатами вашого сервера, наприклад, Бангкока). Розташуйте елементи на дашборді таким чином, щоб усі компоненти були зручно переглядати.
2. Для створених SNMP-даних налаштуйте тригери визначення критичних подій. Порогові значення для критичних подій оберіть на власний розсуд.
3. Створіть у Telegram бот zabbixGNbot, де G – числова частина імені групи, а N – номер варіанту, налаштуйте та протестуйте Media Type для Telegram. Створіть користувача, прив'яжіть його до цього медіа-типу, налаштуйте відправку сповіщень для подій хоста та перевірте роботу сповіщень.

Звіт має містити:

- лістинг використаних команд;
- короткий опис редагування файлів конфігурації;
- скріншоти налаштувань та підключень.

Додаток 16.1.

Створення та видалення Telegram-бота для використання у сповіщеннях.

Для створення Telegram-бота - входимо у Telegram і знаходимо бота BotFather. Надсилаємо команду /start , та виконуємо команду /newbot. Вводимо ім’я бота (наприклад, Zabbix-G-N Alerts Bot) та унікальне ім’я для бота, яке закінчується на bot (наприклад, zabbixGNbot).

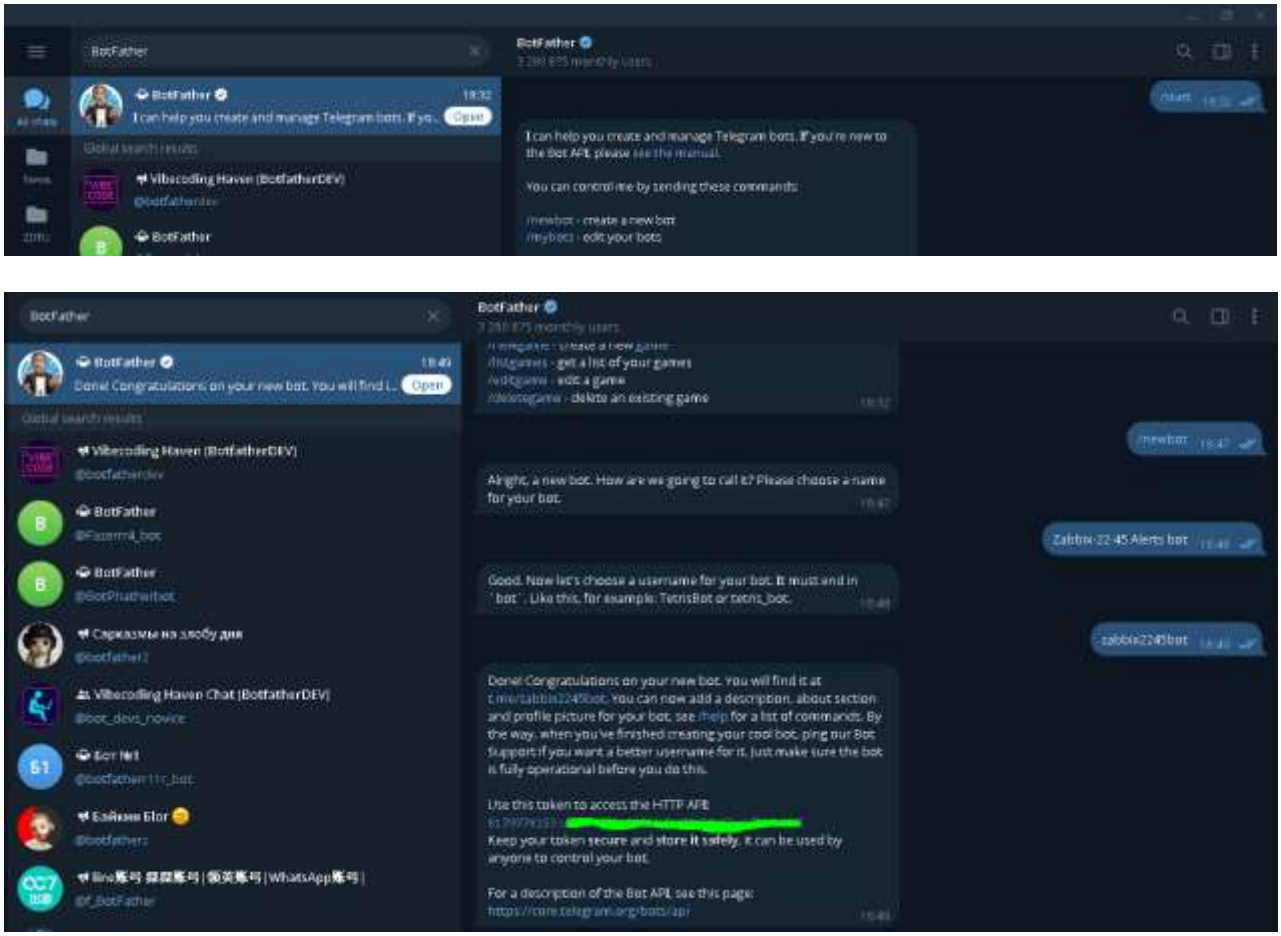


Рис. 16.14. Створення Telegram бота zabbix2240bot.

Якщо дані у діалозі телеграм введено коректно, BotFather надасть токен вигляду

1234567891:BAJn9SVRmj5TEBQEHlSxiYmLfg7u2XI

Зверніть увагу на можливість доступу до Menu графічного керування BotFather в нижній частині діалогу. Зберігаємо токен, оскільки він знадобиться для налаштування Zabbix.

Токен Telegram-бота – це фактично "пароль" для доступу до API цього бота. Хто має токен, той може керувати ботом, надсилати та отримувати повідомлення, виконувати API-запити від його імені.
Ризики витоку токена
Якщо ваш токен потрапить у чужі руки, зловмисник може:

- Використовувати вашого бота для розсилки спаму.
- Отримати доступ до ваших чатів (якщо бот зберігає повідомлення).
- Використовувати бота в шахрайських схемах або для автоматизації атак.

Щоб видалити Telegram-бота, створеного через BotFather, входимо у Telegram та знаходимо бота BotFather. Відправляємо команду /start, якщо ще не починали з ним діалог та надсилаємо команду /deletebot. BotFather попросить ввести ім'я бота, який необхідно видалити (наприклад, @zabbix2240bot). Підтверджуємо видалення, дотримуючись інструкцій BotFather. Після цього бот буде видалено, а його токен стане недійсним. Однак Telegram не дозволяє повністю "стерти" бота – його ім'я та дані можуть залишитися на деякий час в базі, але він більше не буде працювати.

Рис. 16.15. Видалення Telegram бота zabbix2240bot.

Якщо ви повністю видалили бота через BotFather, то Telegram відкликає його токен, і він стає недійсним. Навіть якщо хтось мав доступ до токена раніше, використати його більше не можна.

Якщо ви помітили, що токен «мав витік» (наприклад, в коді на GitHub), терміново змініть токен через BotFather. Відкрийте BotFather і надішліть команду /token. Виберіть свого бота та запросіть новий токен. Після цього старий токен стане недійсним. Оновіть код та налаштуйте сервіси, які використовували старий токен.

Після отримання токену та імен службового боту необхідно отримати ID чату. Переходимо до нового боту в Telegram і натискаємо Start.

Знаходимо chat ID за допомогою наступної URL-адреси в браузері:

https://api.telegram.org/bot<TOKEN>/getUpdates

Замість <TOKEN> підставляємо токен вашого бота типу ***1234567891:BAJn9SVRmj5TEBQEHlSxiYmLfg7u2XI***

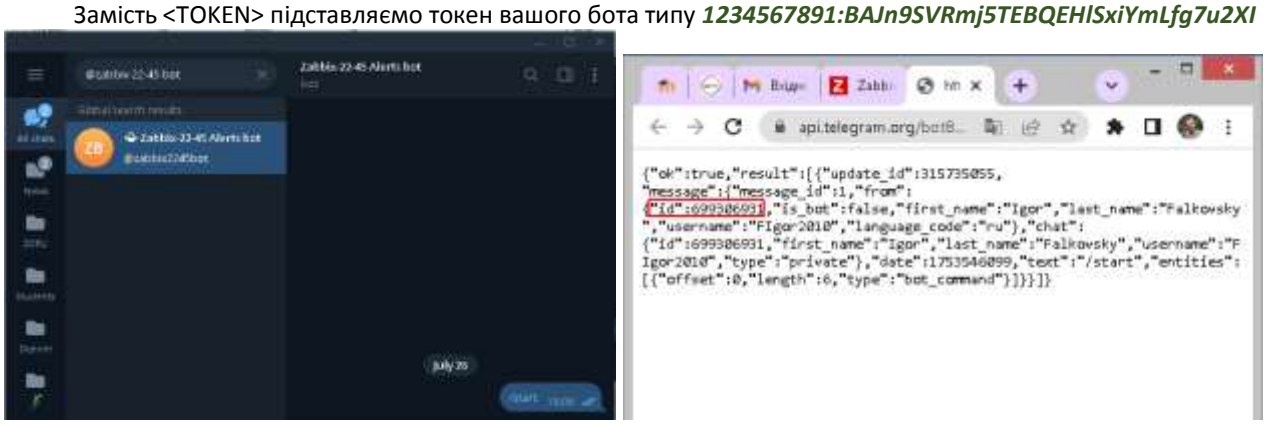


Рис. 16.16. Отримання ID чату для бота zabbix2245bot.

Відповідно до рис.16.16 id чату для бота zabbix2245bot має значення 699306931. Налаштування у телеграм виконано. Для налаштувань сповіщень знадобиться токен та id чату. Остання перевірка – запуск у браузері перевірочного коду

https://api.telegram.org/bot<BOT_TOKEN>/sendMessage?chat_id=<CHAT_ID>&text=Test

де BOT_TOKEN та CHAT_ID токен та id чату створеного боту. У Telegram з'явиться відгук "Test" (рис.16.17).

Рис. 16.17. Тестування бота zabbix2245bot через браузер.

Корисні посилання

- Zabbix DashBoards.

http://surl.li/hmmceo

- Zabbix Alerts: Setup Zabbix Email Notifications & Escalations.

https://bestmonitoringtools.com/zabbix-alerts-setup-zabbix-email-notifications-escalations/

- Zabbix Media Types.

https://www.zabbix.com/documentation/current/en/manual/config/notifications/media

- Zabbix + Telegram.

https://www.zabbix.com/integrations/telegram

- Zabbix – Сповіщення про події в Telegram bot.

https://itorakul.com.ua/zabbix-spovishhennya-pro-podiyi-v-telegram-bot/

- Zabbix-Notification-Telegram.

https://github.com/xxsokolov/Zabbix-Notification-Telegram