Name: Bose, Vincent Bryan	Date Performed: 28/10/2024
Course/Section: CPE31S2	Date Submitted: 28/10/2024
Instructor: Engr. Robin Valenzuela	Semester and SY: 2024-2025

Activity 9: Install, Configure, and Manage Performance Monitoring tools

1. Objectives

Create and design a workflow that installs, configure and manage enterprise performance tools using Ansible as an Infrastructure as Code (IaC) tool.

2. Discussion

Performance monitoring is a type of monitoring tool that identifies current resource consumption of the workload, in this page we will discuss multiple performance monitoring tool.

Prometheus

Prometheus fundamentally stores all data as timeseries: streams of timestamped values belonging to the same metric and the same set of labeled dimensions. Besides stored time series, Prometheus may generate temporary derived time series as the result of queries. Source: Prometheus - Monitoring system & time series database

Cacti

Cacti is a complete network graphing solution designed to harness the power of RRDTool's data storage and graphing functionality. Cacti provides a fast poller, advanced graph templating, multiple data acquisition methods, and user management features out of the box. All of this is wrapped in an intuitive, easy to use interface that makes sense for LAN-sized installations up to complex networks with thousands of devices. Source: Cacti® - The Complete RRDTool-based Graphing Solution

3. Tasks

1. Create a playbook that installs Prometheus in both Ubuntu and CentOS. Apply the concept of creating roles.
2. Describe how you did step 1. (Provide screenshots and explanations in your report. Make your report detailed such that it will look like a manual.)
3. Show an output of the installed Prometheus for both Ubuntu and CentOS.
4. Make sure to create a new repository in GitHub for this activity.

4. Output (screenshots and explanations)

STEP 1.

Just like in the HOA8.1, create a directory named “roles”. Under it make a directory named “base” and “workstations”, under it create a directory named “tasks” for each, and lastly create a “main.yml” for each tasks directory.

```
vbbose@workstation:~/BOSE-HOA-9.1$ ls
ansible.cfg  inventory  prometheus.service.j2  promiteyus.yml  README.md  roles
```

```
vbbose@workstation:~/BOSE-HOA-9.1$ cd roles
vbbose@workstation:~/BOSE-HOA-9.1/roles$ ls
base  workstations
vbbose@workstation:~/BOSE-HOA-9.1/roles$ cd base
vbbose@workstation:~/BOSE-HOA-9.1/roles/base$ ls
tasks
vbbose@workstation:~/BOSE-HOA-9.1/roles/base$ cd tasks
vbbose@workstation:~/BOSE-HOA-9.1/roles/base/tasks$ ls
main.yml
vbbose@workstation:~/BOSE-HOA-9.1/roles/base/tasks$ cd ../..
vbbose@workstation:~/BOSE-HOA-9.1/roles$ cd workstations
vbbose@workstation:~/BOSE-HOA-9.1/roles/workstations$ ls
tasks
vbbose@workstation:~/BOSE-HOA-9.1/roles/workstations$ cd tasks
vbbose@workstation:~/BOSE-HOA-9.1/roles/workstations/tasks$ ls
main.yml
```

STEP 2.

For the main.yml in the base directory, copy this code.

```
---
- name: Update package cache
  apt:
    update_cache: yes
  when: ansible_distribution == "Ubuntu"

- name: Update package cache
  yum:
    update_cache: yes
    use_backend: dnf4
    name: '*'
    state: latest
  when: ansible_distribution == "CentOS"
```

For the main.yml in the workstations directory, copy this code.

```yaml
---
  - name: Add GPG key for the Prometheus repository (Ubuntu)
    apt_key:
      url: https://packages.grafana.com/gpg.key
      state: present
    when: ansible_distribution == "Ubuntu"

  - name: Add Prometheus APT repository (Ubuntu)
    apt_repository:
      repo: deb https://packages.grafana.com/oss/deb stable main
      state: present
      filename: grafana
    when: ansible_distribution == "Ubuntu"

  - name: Install Prometheus (Ubuntu)
    apt:
      name: prometheus
      state: present
    when: ansible_distribution == "Ubuntu"

  - name: Enable and start Prometheus service (Ubuntu)
    service:
      name: prometheus
      enabled: yes
      state: started
    when: ansible_distribution == "Ubuntu"

  - name: Open port 9090 for Prometheus (Ubuntu)
    ufw:
      rule: allow
      port: 9090
      proto: tcp
      state: enabled
    when: ansible_distribution == "Ubuntu"
```

```yaml
  - name: Download Prometheus(CentOS)
    get_url:
      url: "https://github.com/prometheus/prometheus/releases/download/v3.0.0-beta.1/
prometheus-3.0.0-beta.1.linux-amd64.tar.gz"
      dest: /tmp/prometheus.tar.gz
    when: ansible_distribution == "CentOS"

  - name: Extract Prometheus (CentOS)
    ansible.builtin.shell: tar -zxvf /tmp/prometheus.tar.gz -C /tmp/
    when: ansible_distribution == "CentOS"

  - name: Create Prometheus user (CentOS)
    ansible.builtin.user:
      name: prometheus
      state: present
    when: ansible_distribution == "CentOS"

  - name: Create Prometheus directory (CentOS)
    ansible.builtin.file:
      path: /opt/prometheus
      state: directory
    when: ansible_distribution == "CentOS"

  - name: Set ownership and permissions for Prometheus (CentOS)
    ansible.builtin.file:
      path: /opt/prometheus
      owner: prometheus
      group: prometheus
      mode: '0755'
    when: ansible_distribution == "CentOS"

  - name: Create Prometheus service file (CentOS)
    ansible.builtin.template:
      src: prometheus.service.j2
      dest: /etc/systemd/system/prometheus.service
    when: ansible_distribution == "CentOS"


  - name: Reload systemd (CentOS)
    ansible.builtin.command: systemctl daemon-reload
    when: ansible_distribution == "CentOS"

  - name: Start Prometheus service (CentOS)
    ansible.builtin.service:
      name: prometheus
      enabled: yes
      state: started
    when: ansible_distribution == "CentOS"

  - name: Open Firewall Port for Prometheus (CentOS)
    ansible.builtin.shell: firewall-cmd --add-port=9090/tcp --permanent
    when: ansible_distribution == "CentOS"

  - name: Ensure Firewall Rule Reloaded (CentOS)
    ansible.builtin.shell: firewall-cmd --reload
    when: ansible_distribution == "CentOS"

  - name: Enable Prometheus on system boot (CentOS)
    ansible.builtin.service:
      name: prometheus
      enabled: yes
    when: ansible_distribution == "CentOS"
```

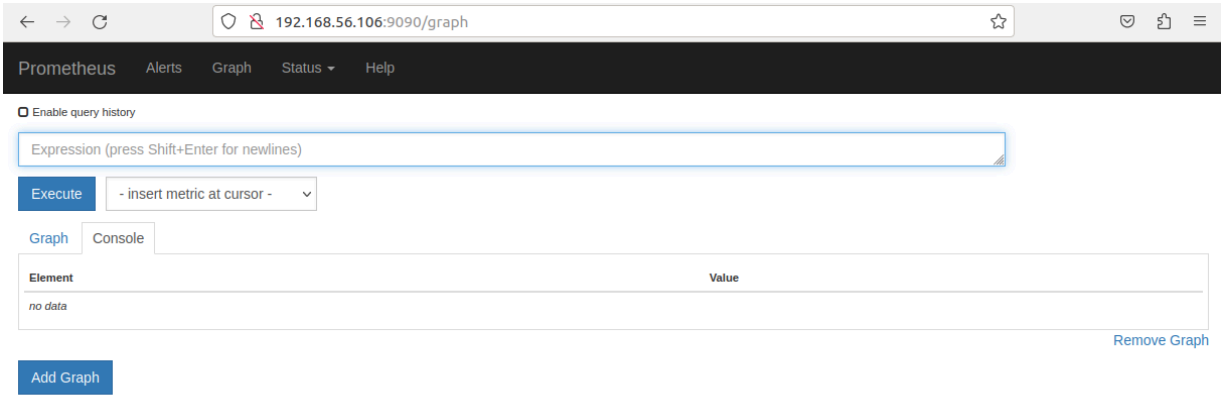
STEP 3. Create a prometheus.service.j2

```
[Unit]
Description=Prometheus Server
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/opt/prometheus/prometheus-2.30.0.linux-amd64/prometheus --config.file=/opt/prometheus/
prometheus-2.30.0.linux-amd64/prometheus.yml
Restart=always

[Install]
WantedBy=multi-user.target
```

STEP 4. Create a prometheus playbook so it can run.

```
  GNU nano 2.9.3                                          promiteyus.yml

---
  - hosts: all
    become: true
    pre_tasks:

    - name: update repository index (CentOS)
      tags: always
      yum:
        update_cache: yes
        use_backend: dnf4
      changed_when: false
      when: ansible_distribution == "CentOS"

    - name: install updates (Ubuntu)
      tags: always
      apt:
        update_cache: yes
      changed_when: false
      when: ansible_distribution == "Ubuntu"

  - hosts: all
    become: true
    roles:
      - base

  - hosts: workstations
    become: true
    roles:
      - workstations
```

```
TASK [workstations : Create Prometheus user (CentOS)] ***************************************************************
skipping: [192.168.56.106]
ok: [192.168.56.113]

TASK [workstations : Create Prometheus directory (CentOS)] **********************************************************
skipping: [192.168.56.106]
ok: [192.168.56.113]

TASK [workstations : Set ownership and permissions for Prometheus (CentOS)] *****************************************
skipping: [192.168.56.106]
ok: [192.168.56.113]

TASK [workstations : Create Prometheus service file (CentOS)] *******************************************************
skipping: [192.168.56.106]
ok: [192.168.56.113]

TASK [workstations : Reload systemd (CentOS)] ***********************************************************************
skipping: [192.168.56.106]
changed: [192.168.56.113]

TASK [workstations : Start Prometheus service (CentOS)] *************************************************************
skipping: [192.168.56.106]
changed: [192.168.56.113]

TASK [workstations : Open Firewall Port for Prometheus (CentOS)] ****************************************************
skipping: [192.168.56.106]
changed: [192.168.56.113]

TASK [workstations : Ensure Firewall Rule Reloaded (CentOS)] ********************************************************
skipping: [192.168.56.106]
changed: [192.168.56.113]

TASK [workstations : Enable Prometheus on system boot (CentOS)] *****************************************************
skipping: [192.168.56.106]
ok: [192.168.56.113]

PLAY RECAP **********************************************************************************************************
192.168.56.106             : ok=10   changed=1    unreachable=0    failed=0    skipped=13   rescued=0    ignored=0
192.168.56.113             : ok=16   changed=6    unreachable=0    failed=0    skipped=7    rescued=0    ignored=0
```



This should be the output of the playbook

Reflections:

Answer the following:

1. What are the benefits of having a performance monitoring tool?
    - Using a performance monitoring tool in an Ubuntu playbook can greatly improve how IT systems are managed. First, it helps keep an eye on how the system is performing in real-time, allowing administrators to spot problems

before they cause issues. This makes it easier to optimize resources and decide when to upgrade or adjust services. The tools can be set up automatically across many servers, ensuring everything is consistent and easy to deploy.

These tools also come with alerts that notify admins when performance drops or when resources are running low, enabling quicker responses to problems. By looking at past performance data, teams can better plan for future needs and troubleshoot issues more easily. They can also help identify security threats if something unusual happens with performance.

Conclusions:

In this activity, creating and designing a workflow for installing, configuring, and managing enterprise performance tools using Ansible as an Infrastructure as Code (IaC) solution helps automate and simplify the performance monitoring process within an organization. This approach not only ensures consistent and efficient deployment across multiple environments but also facilitates better system management through automated configurations. By leveraging Ansible, organizations can enhance their ability to monitor system health, quickly address performance issues, and make informed decisions based on real-time data. Ultimately, this workflow aims to improve operational efficiency, increase productivity, and support a proactive approach to IT management, leading to more reliable and scalable enterprise systems.