Name: Khelvin P. Nicolas	Date Performed: 16/10/2024
Course/Section: CPE 212 - CPE31S2	Date Submitted: 16/10/2024
Instructor: Engr. Robin Valenzuela	Semester and SY: 1st Sem - 3rd Yr.

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 tools.

Prometheus

Prometheus fundamentally stores all data as time series: 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)

Phase 1: Setting up working environment for Prometheus installation

To install Prometheus in our managed nodes, we must first have a working environment which we can use to install any application that we might use, whether it is a real corporate environment or not. In this activity, we created a GitHub repository to save all the configurations and playbooks that will be used, it also includes the inventory file.

GitHub Repository for Activity 9: https://github.com/KHLVN/CPE212_Activity9

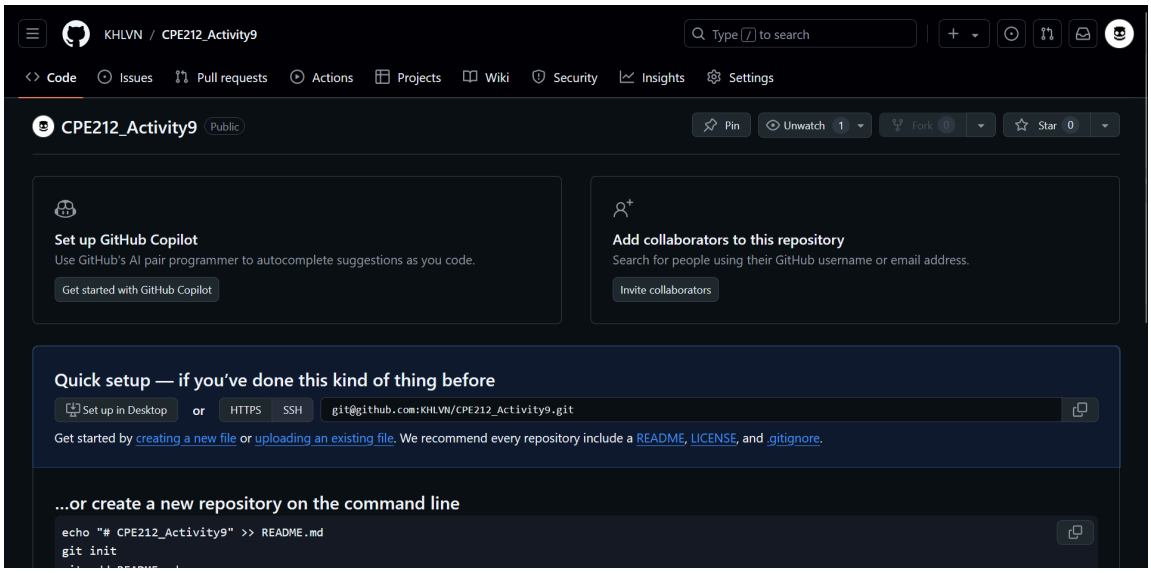

Figure 1.1: Create GitHub repository

Inside the workstation, we cloned our newly created repository. We will perform the activity inside of this directory so we can commit and push it easily in the remote repository later.

```
punopaughey@workstation:~$ git clone git@github.com:KHLVN/CPE212_Activity9.git
Cloning into 'CPE212_Activity9'...
warning: You appear to have cloned an empty repository.
punopaughey@workstation:~$ cd *9
```

Figure 1.2: Cloning created repo in control node

We created directories for specific roles that we want to assign to our managed nodes, including `**db_servers**` for database management, `**file_servers**` for file sharing and FTP, and `**web_servers**` for web hosting. Each directory contains a `**tasks**` subdirectory where we can place a playbook to perform tasks for the corresponding role.

```
punopaughey@workstation:~/CPE212_Activity9$ cp ~/*8/ansible.cfg .
punopaughey@workstation:~/CPE212_Activity9$ cp ~/*8/inventory .
punopaughey@workstation:~/CPE212_Activity9$ cp ~/*8/routine.yml .
punopaughey@workstation:~/CPE212_Activity9$ cp -r ~/*8/roles .
punopaughey@workstation:~/CPE212_Activity9$ ls
ansible.cfg  inventory  roles  routine.yml
punopaughey@workstation:~/CPE212_Activity9$
```

Figure 1.3: Copying the contents from last activity

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

  - name: update repository index (CentOS)
    dnf:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "CentOS"

  - name: update repository index (Ubuntu)
    apt:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "Ubuntu"

- hosts: workstations
  become: true
  roles:
    - workstations

- hosts: db_servers
  become: true
  roles:
    - db_servers

- hosts: web_servers
  become: true
  roles:
    - web_servers

- hosts: file_servers
  become: true
  roles:
    - file_servers
```

Figure 1.4: Same initial routine.yml from last activity

Figure 1.5: Creating initial main.yml file inside all roles' directories

By mapping our network environment, we can see the following nodes:

- **workstation (control node)**
- **server1 (managed node)**
- **server2 (managed node)**
- **server3 (managed node)**
- **centos (managed node)**

Servers 1, 2, and 3 uses Ubuntu distribution while **centos** uses CentOS distribution

If you remember the last activity, this is how we configured the managed nodes and assigned roles, but for reference, this is the inventory file that we used in Activity 8.

```
[workstations]
workstation

[web_servers]
server1

[db_servers]
centos   ansible_user=khlvn

[file_servers]
server2
server3
```

Phase 2: Installing Prometheus using Ansible Playbook

After setting up the environment, we are now ready to proceed to the next step which is to install Prometheus availability monitoring tool in our remote servers. Create a playbook that will install Prometheus on both distributions (CentOS and Ubuntu). After the installation of Prometheus is complete, we should also include another task in the playbook that enables the Prometheus service for use in our web browsers. Refer to the sample code below to properly install Prometheus on our systems.

```
---
 - name: Update Repository Index (db_servers)
   package:
         update_cache: yes
   changed_when: false

 - name: Open port 9090
   ufw:
         rule: allow
         port: 9090
         proto: tcp
```

```
        state: enabled

 - name: Allow Prometheus for Firewall
   firewalld:
        port: 9090/tcp
        permanent: yes
        state: enabled

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

 - name: Install Prometheus (CentOS)
   unarchive:
        src:
https://github.com/prometheus/prometheus/releases/download/v2.30.0/prometheus-
2.30.0.linux-amd64.tar.gz
        dest: /usr/local/bin
        remote_src: yes
        mode: 0755
        owner: root
        group: root
   when: ansible_distribution == "CentOS"

 - name: Copy Prometheus binaries
   copy:
        src: /usr/local/bin/prometheus-2.30.0.linux-amd64/prometheus
        dest: /usr/local/bin/prometheus
        mode: 0755
        remote_src: yes
   when: ansible_distribution == "CentOS"

 - name: Copy Promtool binaries
   copy:
        src: /usr/local/bin/prometheus-2.30.0.linux-amd64/prometheus
        dest: /usr/local/bin/promtool
        mode: 0755
        remote_src: yes
   when: ansible_distribution == "CentOS"

 - name: Create Prometheus directories
   file:
        path: "{{ item }}"
```

```
        state: directory
  loop:
        - /etc/prometheus
        - /var/lib/prometheus
  when: ansible_distribution == "CentOS"

- name: Copy prometheus.yml to /etc/prometheus
  command: cp /usr/local/bin/prometheus-2.30.0.linux-amd64/prometheus.yml
/etc/prometheus
  when: ansible_distribution == "CentOS"

- name: Copy consoles directory to /etc/prometheus
  command: cp -r /usr/local/bin/prometheus-2.30.0.linux-amd64/consoles
/etc/prometheus
  when: ansible_distribution == "CentOS"

- name: Copy console_libraries directory to /etc/prometheus
  command: cp -r /usr/local/bin/prometheus-2.30.0.linux-amd64/console_libraries
/etc/prometheus
  when: ansible_distribution == "CentOS"

- name: Create prometheus.service file
  copy:
        dest: /etc/systemd/system/prometheus.service
        content: |
        [Unit]
        Description=Prometheus
        Wants=network-online.target
        After=network-online.target

        [Service]
        User=root
        Group=root
        Type=simple
        ExecStart=/usr/local/bin/prometheus \
                --config.file /etc/prometheus/prometheus.yml \
                --storage.tsdb.path /var/lib/prometheus \
                --web.console.templates=/etc/prometheus/consoles \
                --web.console.libraries=/etc/prometheus/console_libraries \

        [Install]
        WantedBy=multi-user.target
  when: ansible_distribution == "CentOS"

- name: Reload systemd
```

```
  command: systemctl daemon-reload
  when: ansible_distribution == "CentOS"

- name: Start Prometheus Service
  systemd:
        name: prometheus
        enabled: yes
        state: started
  when: ansible_distribution == "CentOS"
```

Figure 2.1: Create main.yml file to install Prometheus in both Ubuntu and CentOS

After writing the code above and saving it as a YAML file, we should now run the playbook using the command ***ansible-playbook --ask-become-pass <yourplaybookname>.yml***.

```
TASK [web_servers : Copy prometheus.yml to /etc/prometheus] ********************
skipping: [server1]

TASK [web_servers : Copy consoles directory to /etc/prometheus] ****************
skipping: [server1]

TASK [web_servers : Copy console_libraries directory to /etc/prometheus] *******
skipping: [server1]

TASK [web_servers : Create prometheus.service file] ****************************
skipping: [server1]

TASK [web_servers : Reload systemd] ********************************************
skipping: [server1]

TASK [web_servers : Start Prometheus Service] **********************************
skipping: [server1]

TASK [web_servers : Start Prometheus Service (Ubuntu)] *************************
ok: [server1]
[WARNING]: Could not match supplied host pattern, ignoring: file_servers

PLAY [file_servers] ************************************************************
skipping: no hosts matched

PLAY RECAP *********************************************************************
centos                     : ok=15   changed=4    unreachable=0    failed=0    skipped=
3    rescued=0    ignored=0
server1                    : ok=6    changed=1    unreachable=0    failed=0    skipped=
11   rescued=0    ignored=0
```

Figure 2.2: Running the ansible playbook

After running the playbook, as shown in figure 1.8, we can see that Prometheus was successfully installed along with its directories and service properly set, using only the Ansible playbook.

Now try putting ***server1:9090*** and ***centos:9090*** in the browser of the control node, it should display the dashboard page for Prometheus.

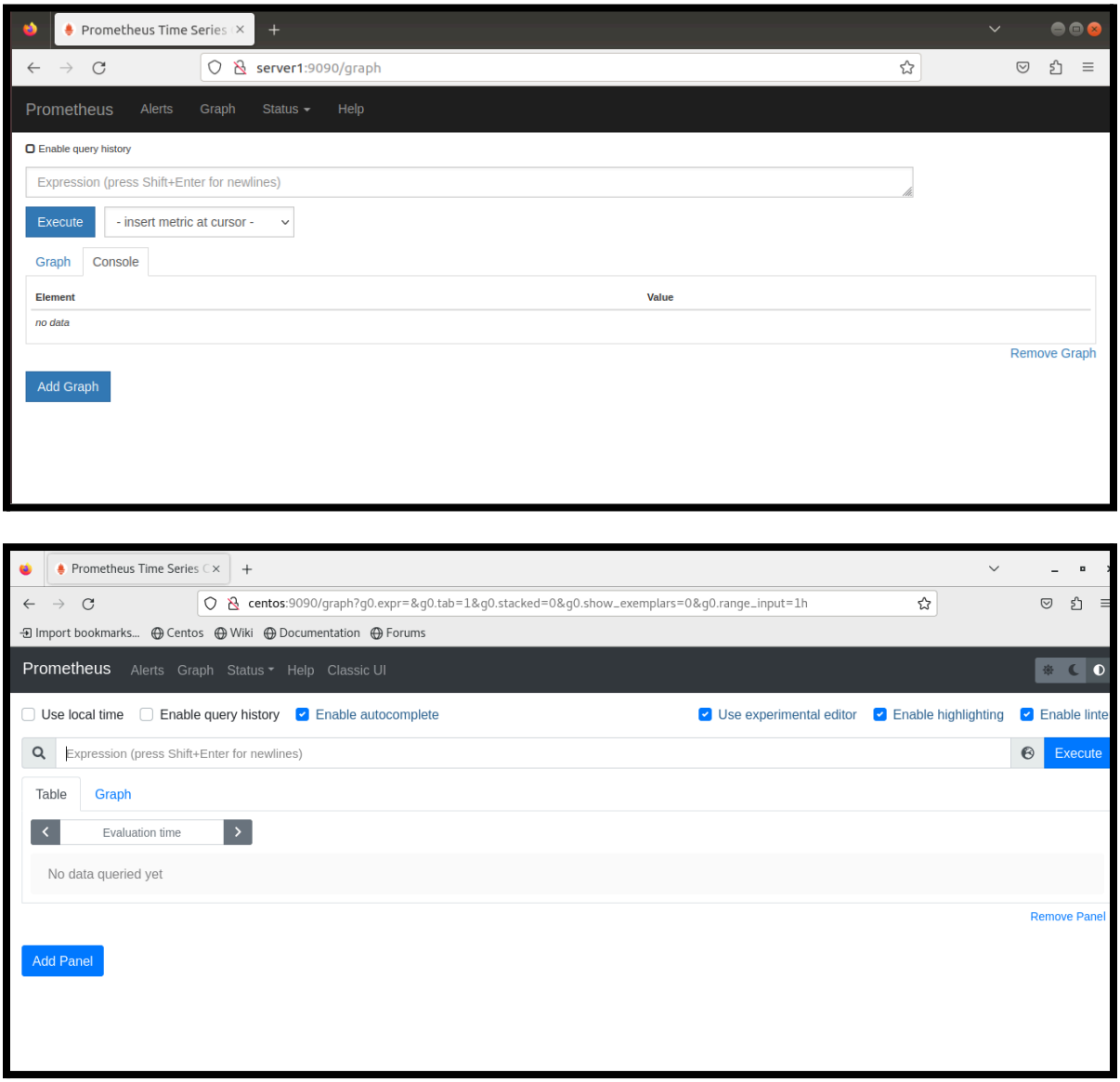

Figure 2.3: Testing Prometheus in Firefox for both Ubuntu and CentOS

Now that Prometheus has successfully installed in our remote servers, we must now push the created playbooks and directories in our remote repository to be used for reference later.

```
punopaughey@workstation:~/CPE212_Activity9$ ls
ansible.cfg  inventory  main.yml  roles  routine.yml
punopaughey@workstation:~/CPE212_Activity9$ git commit -m "Act 9"
[master (root-commit) b568138] Act 9
 5 files changed, 60 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 inventory
 create mode 100644 main.yml
 create mode 100644 roles/workstations/tasks/main.yml
 create mode 100644 routine.yml
punopaughey@workstation:~/CPE212_Activity9$ git push origin master
Counting objects: 9, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 974 bytes | 974.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0)
To github.com:KHLVN/CPE212_Activity9.git
 * [new branch]      master -> master
punopaughey@workstation:~/CPE212_Activity9$
```

Figure 2.4: Pushing into the remote repository

Reflections:

Answer the following:

1. **What are the benefits of having a performance monitoring tool?**
   - A performance monitoring tool is much more than just a technical tool. it is a vital part of how an organization runs day-to-day. It gives teams the insight they need to keep their IT systems running smoothly, tackle challenges head-on, and make improvements that benefit users. By focusing on system health and optimizing performance, the organizations can better reach their objective. Ultimately, it helps create a more responsive and efficient environment that supports both the team and the users they serve.

Conclusions:

- In the last activity, we installed Nagios, which is used as an availability monitoring tool. This activity now instructs us to install a performance management tool named Prometheus. We performed the same tasks as in the last activity, but this time we configured the services to run, added extra steps to ensure that the service is in an active state, and used Prometheus as a localhost/web service.