Name: Kazuki A. Ogata	**Date Performed:** October 24, 2023
Course/Section: CPE 232 - CPE31S5	**Date Submitted:** October 25, 2023
Instructor: Engr. Roman Richard	**Semester and SY:** 1st semester S.Y 2023-2024

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

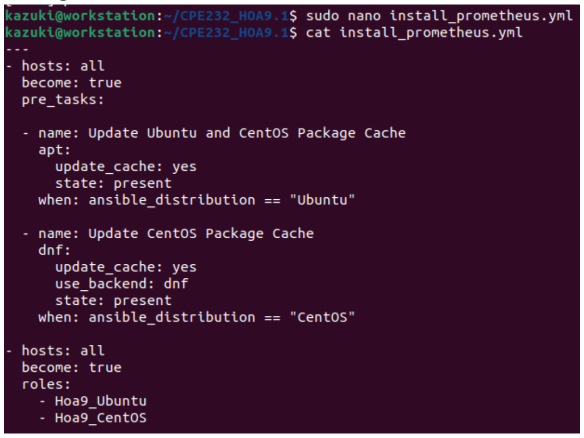
Cacti
Cacti is a complete network graphing solution designed to harness the power of RRDTool's data storage and graphing functionality. Cacti provides a fast poller, advanced graph templating, multiple data acquisition methods, and user management features out of the box. All of this is wrapped in an 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

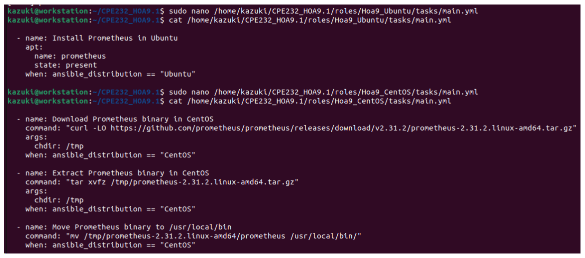
1. Create a playbook that installs Prometheus in both Ubuntu and CentOS. Apply the concept of creating roles.
2. Describe how you did step 1. (Provide screenshots and explanations in your report. Make your report detailed such that it will look like a manual.)
3. Show an output of the installed Prometheus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

4. Output (screenshots and explanations)

1. Create a playbook that installs Prometheus in both Ubuntu and CentOS. Apply the concept of creating roles.



```
kazuki@workstation:~/CPE232_HOA9.1$ sudo nano install_prometheus.yml
kazuki@workstation:~/CPE232_HOA9.1$ cat install_prometheus.yml
---
- hosts: all
  become: true
  pre_tasks:

  - name: Update Ubuntu and CentOS Package Cache
    apt:
      update_cache: yes
      state: present
    when: ansible_distribution == "Ubuntu"

  - name: Update CentOS Package Cache
    dnf:
      update_cache: yes
      use_backend: dnf
      state: present
    when: ansible_distribution == "CentOS"

- hosts: all
  become: true
  roles:
    - Hoa9_Ubuntu
    - Hoa9_CentOS
```

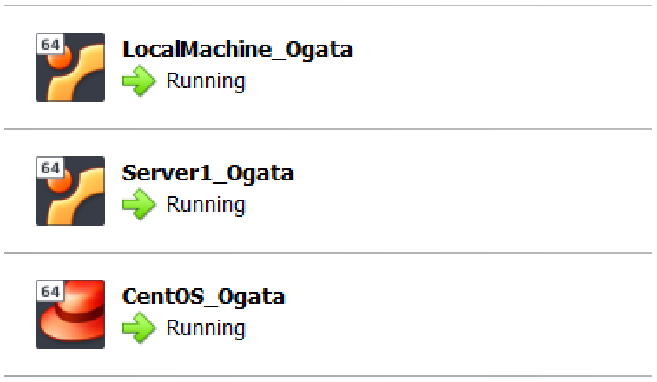


```
kazuki@workstation:~/CPE232_HOA9.1$ sudo nano /home/kazuki/CPE232_HOA9.1/roles/Hoa9_Ubuntu/tasks/main.yml
kazuki@workstation:~/CPE232_HOA9.1$ cat /home/kazuki/CPE232_HOA9.1/roles/Hoa9_Ubuntu/tasks/main.yml

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

kazuki@workstation:~/CPE232_HOA9.1$ sudo nano /home/kazuki/CPE232_HOA9.1/roles/Hoa9_CentOS/tasks/main.yml
kazuki@workstation:~/CPE232_HOA9.1$ cat /home/kazuki/CPE232_HOA9.1/roles/Hoa9_CentOS/tasks/main.yml

  - name: Download Prometheus binary in CentOS
    command: "curl -LO https://github.com/prometheus/prometheus/releases/download/v2.31.2/prometheus-2.31.2.linux-amd64.tar.gz"
    args:
      chdir: /tmp
    when: ansible_distribution == "CentOS"

  - name: Extract Prometheus binary in CentOS
    command: "tar xvfz /tmp/prometheus-2.31.2.linux-amd64.tar.gz"
    args:
      chdir: /tmp
    when: ansible_distribution == "CentOS"

  - name: Move Prometheus binary to /usr/local/bin
    command: "mv /tmp/prometheus-2.31.2.linux-amd64/prometheus /usr/local/bin/"
    when: ansible_distribution == "CentOS"
```

This is my final Playbook file, it contains 2 roles (Ubuntu and CentOS). I applied all of my learnings to the previous activity in this activity. The contents of two roles are also shown.

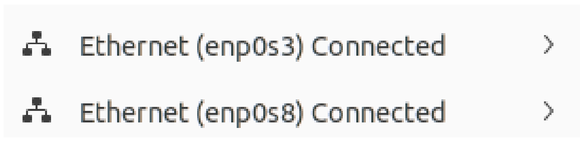
2. Describe how you did step 1. (Provide screenshots and explanations in your report. Make your report detailed such that it will look like a manual.)

Step 1: Open Virtual Machines



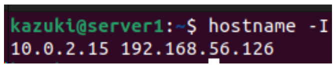
Open Oracle VM Virtual box and start 3 virtual machines, 1 for control node (LocalMachine_Ogata) and 2 for managed nodes for Ubuntu and CentOS (Server2_Ogata and CentOS_Ogata).

Step 2: Check the Internet Connection of Virtual Machines

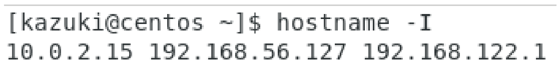


On the local machine and remote servers, type “ping google.com” or click the top right side of your vm to check if the Ethernets are connected (like image above). We are doing this, to avoid possible or future errors. Like what I experience from past activities.

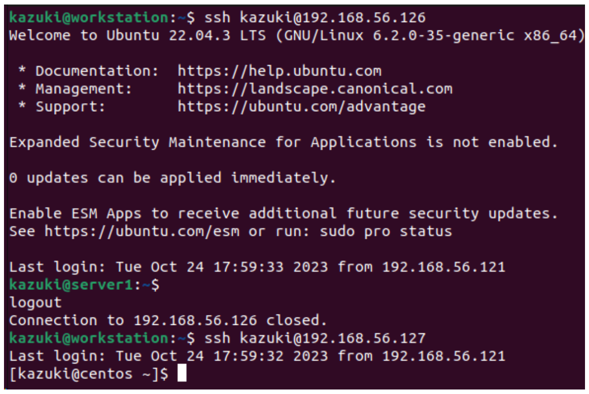
Step 3: Verify Connection



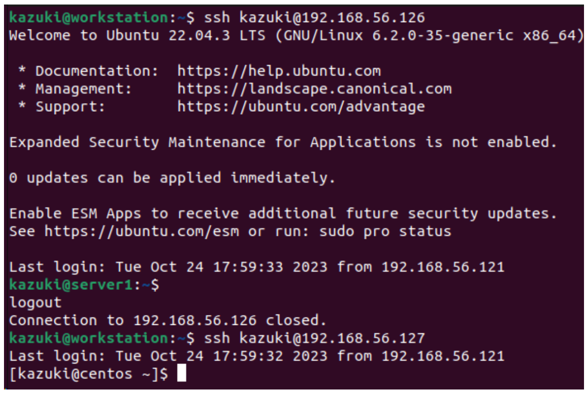
```
kazuki@server1:~$ hostname -I
10.0.2.15 192.168.56.126
```



```
[kazuki@centos ~]$ hostname -I
10.0.2.15 192.168.56.127 192.168.122.1
```



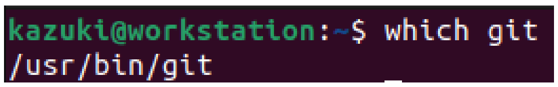
```
kazuki@workstation:~$ ssh kazuki@192.168.56.126
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-35-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Tue Oct 24 17:59:33 2023 from 192.168.56.121
kazuki@server1:~$
logout
Connection to 192.168.56.126 closed.
kazuki@workstation:~$ ssh kazuki@192.168.56.127
Last login: Tue Oct 24 17:59:32 2023 from 192.168.56.121
[kazuki@centos ~]$ 
```



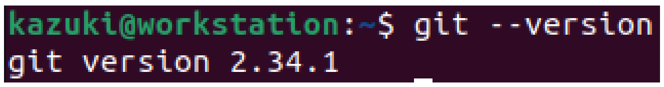
```
kazuki@workstation:~$ ssh kazuki@192.168.56.126
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-35-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Tue Oct 24 17:59:33 2023 from 192.168.56.121
kazuki@server1:~$
logout
Connection to 192.168.56.126 closed.
kazuki@workstation:~$ ssh kazuki@192.168.56.127
Last login: Tue Oct 24 17:59:32 2023 from 192.168.56.121
[kazuki@centos ~]$ 
```

The most important part before starting this activity is checking if there is a connection between control nodes and managed nodes. In this part, we can use *ifconfig* or *hostname -I* to check the IP address and then use *ping* or *ssh user@host* to verify the connections. To check if it is successful you should get the same output shown above.

Step 4: Setup Git

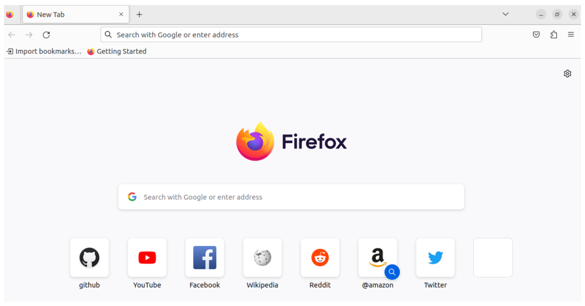


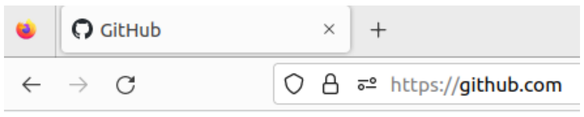
```
kazuki@workstation:~$ which git
/usr/bin/git
```



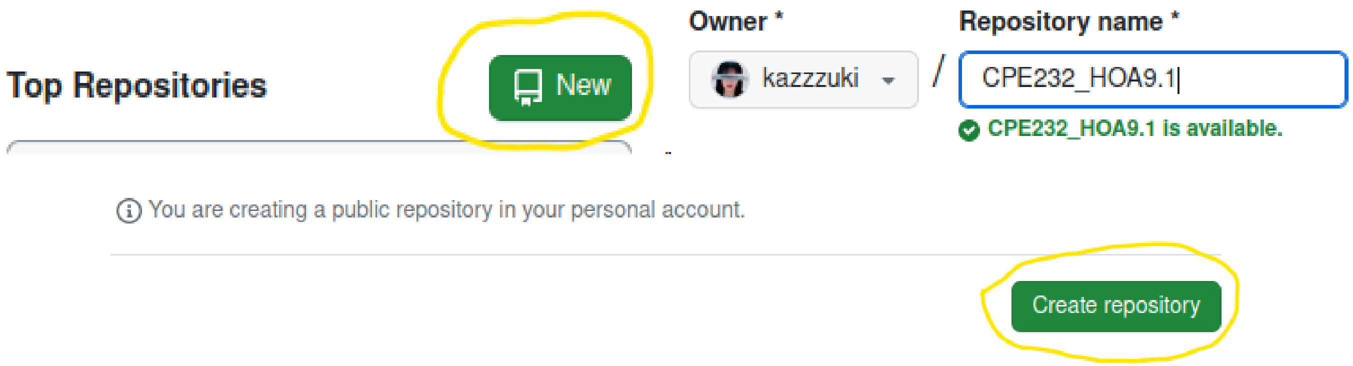
```
kazuki@workstation:~$ git --version
git version 2.34.1
```

On the local machine, use the *which git* command to check if you already have git. If the directory *usr/bin/git* is displayed then you already have git and you don't need to install git. Just in case you still don't have a git you can use the command *sudo apt install git.* You can use *git --version* to check the version.

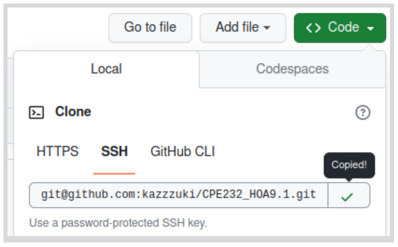




Open any web server available in your workstation and go to www.github.com, if you don't have an account create one by signing up. If you already have an account proceed to the next step.


Top Repositories
New
Owner *
kazzzuki
/
Repository name *
CPE232_HOA9.1
CPE232_HOA9.1 is available.
You are creating a public repository in your personal account.
Create repository

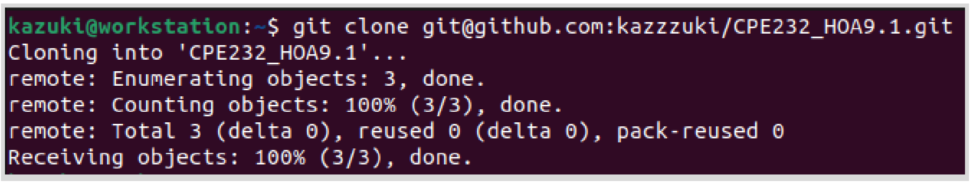
Create a new repository and name it "CPE232_HOA9.1". Check the images above in creating a repository and follow them. I already have an SSH key on my GitHub, if you don't have one yet, you can create a new SSH key on GitHub by going to your profile's settings and then click the "SSH and GPG keys" and click the "New SSH Key". On the local machine, use the command "cat .ssh/id_rsa/pub" and copy its output and then paste it on the GitHub "Key".



```
git@github.com:kazzzuki/CPE232_HOA9.1.git
```

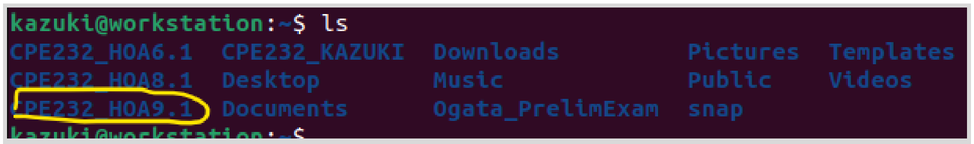
Use a password-protected SSH key.

Clone GitHub repository in local machine, browse your new repository and click the "<> Code" to get the SSH link, copy it.



```
kazuki@workstation:~$ git clone git@github.com:kazzzuki/CPE232_HOA9.1.git
Cloning into 'CPE232_HOA9.1'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

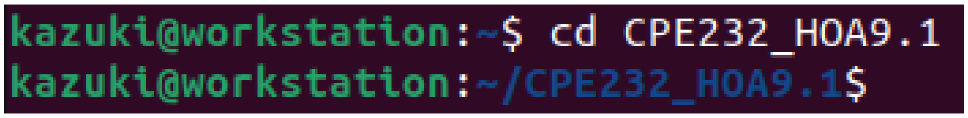
Clone the new repository in your local machine by using the command *git clone* git@github.com:jvtaylar-cpe/CPE232_yourname.git.



```
kazuki@workstation:~$ ls
CPE232_HOA6.1  CPE232_KAZUKI  Downloads        Pictures  Templates
CPE232_HOA8.1  Desktop        Music            Public    Videos
CPE232_HOA9.1  Documents      Ogata_PrelimExam snap
kazuki@workstation:~$
```

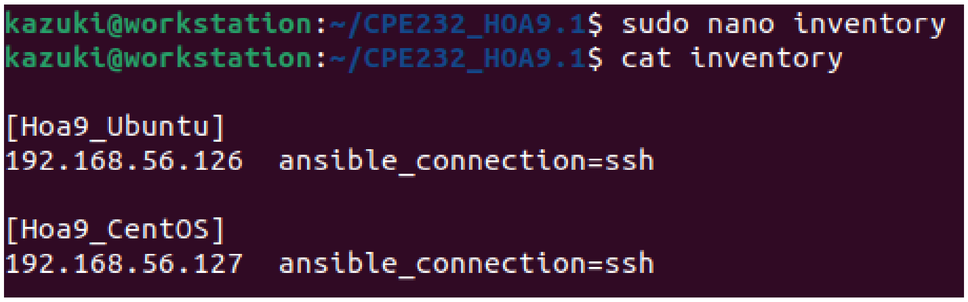
Use the *ls* command to check the cloned repository. You can also personalize your git by using the commands *git config --global user.name "Your Name"* and *git config --global user.email "Your Email"*. Use the *cat ~/.gitconfig* command to verify it.

Step 5: Create a Playbook



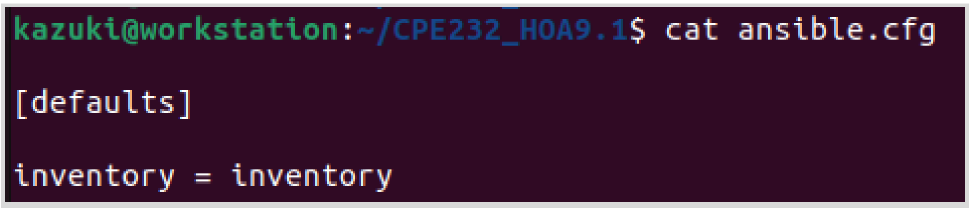
```
kazuki@workstation:~$ cd CPE232_HOA9.1
kazuki@workstation:~/CPE232_HOA9.1$
```

On the local machine, use the command *cd* command to go to the new repository directory. So we can proceed in creating files needed for this activity.



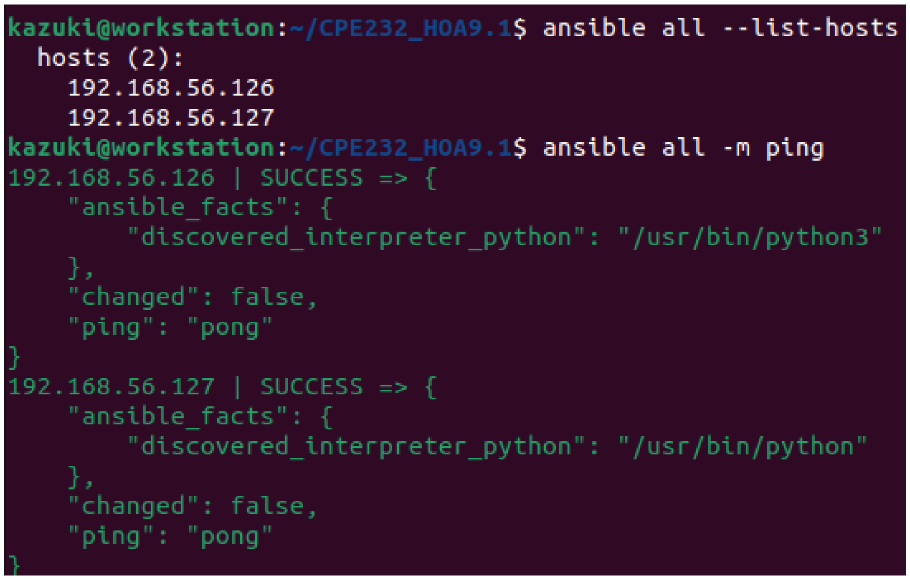
```
kazuki@workstation:~/CPE232_HOA9.1$ sudo nano inventory
kazuki@workstation:~/CPE232_HOA9.1$ cat inventory

[Hoa9_Ubuntu]
192.168.56.126   ansible_connection=ssh

[Hoa9_CentOS]
192.168.56.127   ansible_connection=ssh
```

Create an inventory file. Inside the inventory put the managed nodes IP addresses. I created two groups (Hoa9_Ubuntu and Hoa9_CentOS) and put the IP address of remote servers and added the "ansible_connection=ssh" to specify the connection to be used in connecting to the remote servers. I created a group so it will be easily called in roles later. I used the *cat* command to show the content of the inventory file.



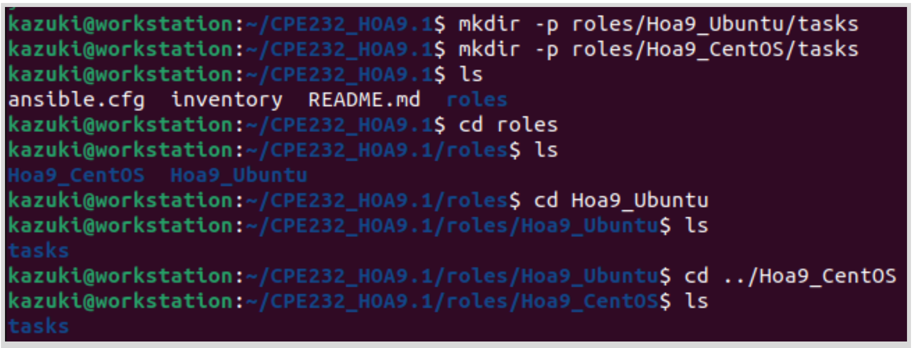
```
kazuki@workstation:~/CPE232_HOA9.1$ cat ansible.cfg

[defaults]

inventory = inventory
```

Create an ansible.cfg file. I added the default configuration and set the default inventory file that Ansible should use.



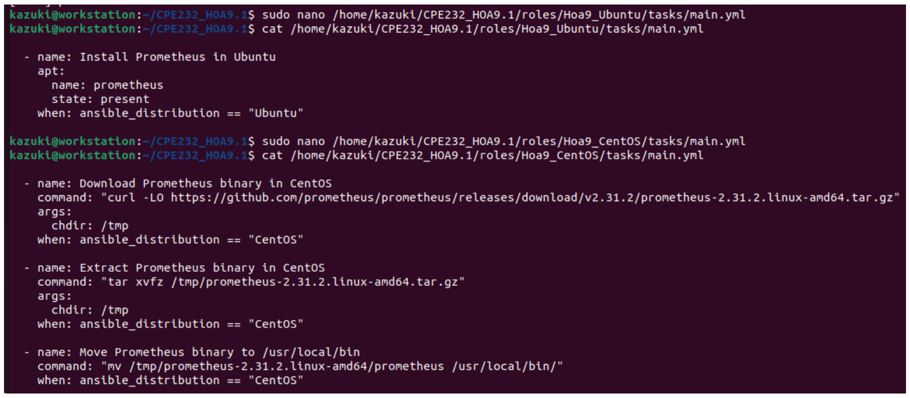
```
kazuki@workstation:~/CPE232_HOA9.1$ ansible all --list-hosts
  hosts (2):
    192.168.56.126
    192.168.56.127
kazuki@workstation:~/CPE232_HOA9.1$ ansible all -m ping
192.168.56.126 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
192.168.56.127 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

Verify the hosts in the inventory. After creating the inventory and ansible.cfg files, we can use the commands *ansible all –list-hosts* and *ansible all -m ping.* You should see the IP address and success pinging in the outputs.



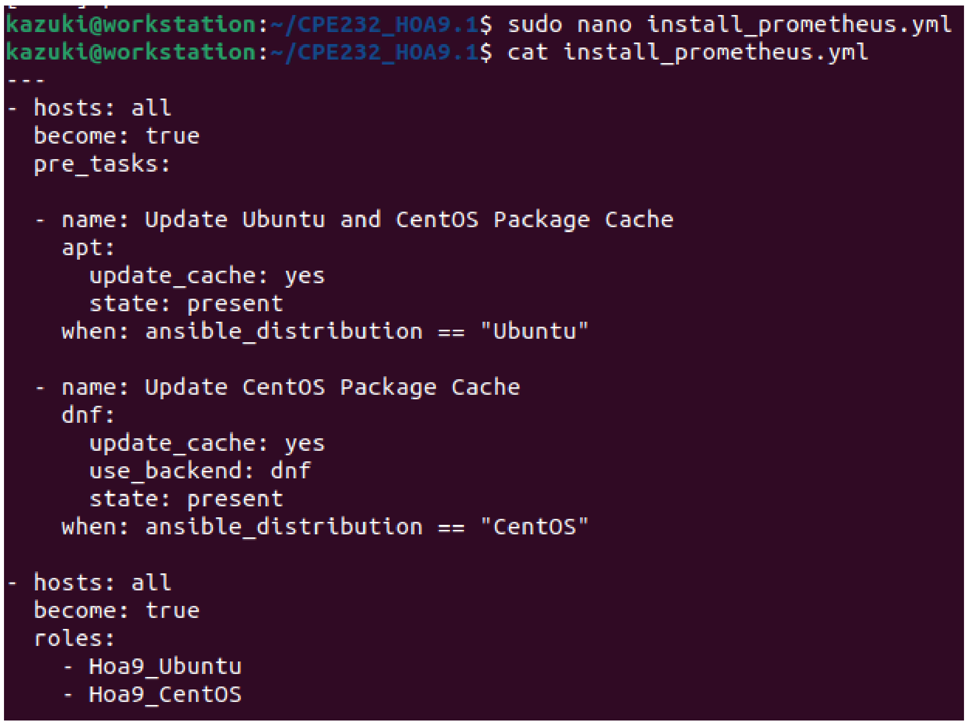
```
kazuki@workstation:~/CPE232_HOA9.1$ mkdir -p roles/Hoa9_Ubuntu/tasks
kazuki@workstation:~/CPE232_HOA9.1$ mkdir -p roles/Hoa9_CentOS/tasks
kazuki@workstation:~/CPE232_HOA9.1$ ls
ansible.cfg  inventory  README.md  roles
kazuki@workstation:~/CPE232_HOA9.1$ cd roles
kazuki@workstation:~/CPE232_HOA9.1/roles$ ls
Hoa9_CentOS  Hoa9_Ubuntu
kazuki@workstation:~/CPE232_HOA9.1/roles$ cd Hoa9_Ubuntu
kazuki@workstation:~/CPE232_HOA9.1/roles/Hoa9_Ubuntu$ ls
tasks
kazuki@workstation:~/CPE232_HOA9.1/roles/Hoa9_Ubuntu$ cd ../Hoa9_CentOS
kazuki@workstation:~/CPE232_HOA9.1/roles/Hoa9_CentOS$ ls
tasks
```

Create directories using the *mkdir* command. I am creating directories where I will put a role. I created Hoa9_Ubuntu and Hoa9_CentOS with tasks directories inside them.



```
kazuki@workstation:~/CPE232_HOA9.1$ sudo nano /home/kazuki/CPE232_HOA9.1/roles/Hoa9_Ubuntu/tasks/main.yml
kazuki@workstation:~/CPE232_HOA9.1$ cat /home/kazuki/CPE232_HOA9.1/roles/Hoa9_Ubuntu/tasks/main.yml

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

kazuki@workstation:~/CPE232_HOA9.1$ sudo nano /home/kazuki/CPE232_HOA9.1/roles/Hoa9_CentOS/tasks/main.yml
kazuki@workstation:~/CPE232_HOA9.1$ cat /home/kazuki/CPE232_HOA9.1/roles/Hoa9_CentOS/tasks/main.yml

  - name: Download Prometheus binary in CentOS
    command: "curl -LO https://github.com/prometheus/prometheus/releases/download/v2.31.2/prometheus-2.31.2.linux-amd64.tar.gz"
    args:
      chdir: /tmp
    when: ansible_distribution == "CentOS"

  - name: Extract Prometheus binary in CentOS
    command: "tar xvfz /tmp/prometheus-2.31.2.linux-amd64.tar.gz"
    args:
      chdir: /tmp
    when: ansible_distribution == "CentOS"

  - name: Move Prometheus binary to /usr/local/bin
    command: "mv /tmp/prometheus-2.31.2.linux-amd64/prometheus /usr/local/bin/"
    when: ansible_distribution == "CentOS"
```

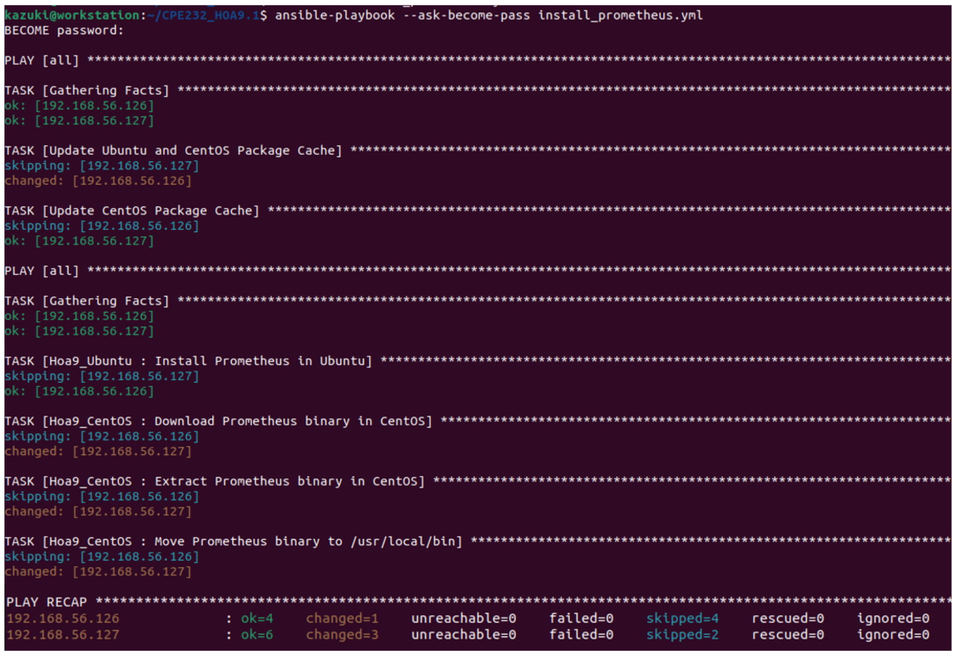
In the main.yml of Hoa9_Ubuntu tasks, I just added the package name of Prometheus and added present state to it so it will install the present Prometheus package in the Ubuntu server. For the main.yml of Hoa9_CentOS tasks, I added tasks for downloading, extracting, and moving Prometheus.



```
kazuki@workstation:~/CPE232_HOA9.1$ sudo nano install_prometheus.yml
kazuki@workstation:~/CPE232_HOA9.1$ cat install_prometheus.yml
---
- hosts: all
  become: true
  pre_tasks:

  - name: Update Ubuntu and CentOS Package Cache
    apt:
      update_cache: yes
      state: present
    when: ansible_distribution == "Ubuntu"

  - name: Update CentOS Package Cache
    dnf:
      update_cache: yes
      use_backend: dnf
      state: present
    when: ansible_distribution == "CentOS"

- hosts: all
  become: true
  roles:
    - Hoa9_Ubuntu
    - Hoa9_CentOS
```

This is my playbook. I put a pre-tasks where it updates the package cache of both Ubuntu and CentOS. I added that to make sure that my remote servers are updated and to avoid errors. And then I put the two roles below.

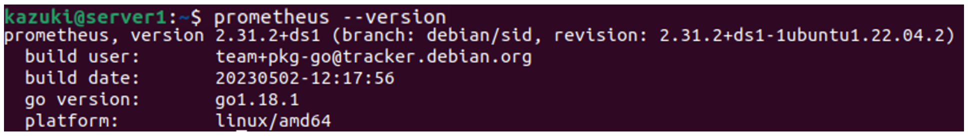
Step 6: Running the Playbook



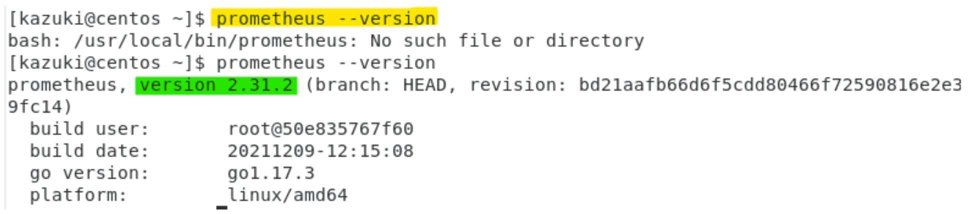
```
kazuki@workstation:~/CPE232_HOA9.1$ ansible-playbook --ask-become-pass install_prometheus.yml
BECOME password:

PLAY [all] ***************************************************************************************

TASK [Gathering Facts] ***************************************************************************
ok: [192.168.56.126]
ok: [192.168.56.127]

TASK [Update Ubuntu and CentOS Package Cache] ****************************************************
skipping: [192.168.56.127]
changed: [192.168.56.126]

TASK [Update CentOS Package Cache] ***************************************************************
skipping: [192.168.56.126]
ok: [192.168.56.127]

PLAY [all] ***************************************************************************************

TASK [Gathering Facts] ***************************************************************************
ok: [192.168.56.126]
ok: [192.168.56.127]

TASK [Hoa9_Ubuntu : Install Prometheus in Ubuntu] ************************************************
skipping: [192.168.56.127]
ok: [192.168.56.126]

TASK [Hoa9_CentOS : Download Prometheus binary in CentOS] ****************************************
skipping: [192.168.56.126]
changed: [192.168.56.127]

TASK [Hoa9_CentOS : Extract Prometheus binary in CentOS] *****************************************
skipping: [192.168.56.126]
changed: [192.168.56.127]

TASK [Hoa9_CentOS : Move Prometheus binary to /usr/local/bin] ************************************
skipping: [192.168.56.126]
changed: [192.168.56.127]

PLAY RECAP ***************************************************************************************
192.168.56.126             : ok=4    changed=1    unreachable=0    failed=0    skipped=4    rescued=0    ignored=0
192.168.56.127             : ok=6    changed=3    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

The output of my playbook when executed shows 0 failed/error. It shows ok, skipped, and changed, meaning all the tasks inside the roles are all successfully executed.

3. Show an output of the installed Prometheus for both Ubuntu and CentOS.



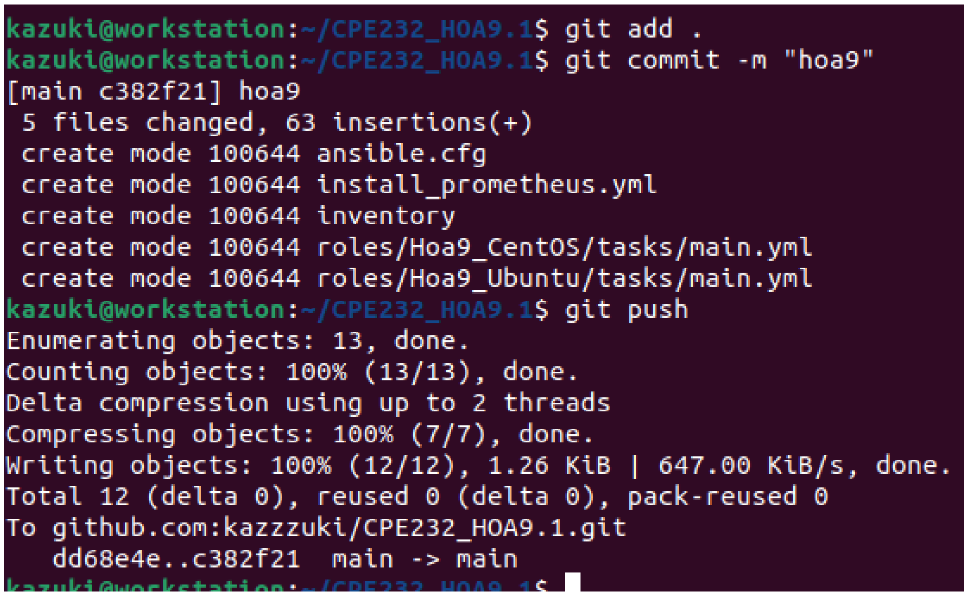
```
kazuki@server1:~$ prometheus --version
prometheus, version 2.31.2+ds1 (branch: debian/sid, revision: 2.31.2+ds1-1ubuntu1.22.04.2)
  build user:       team+pkg-go@tracker.debian.org
  build date:       20230502-12:17:56
  go version:       go1.18.1
  platform:         linux/amd64
```



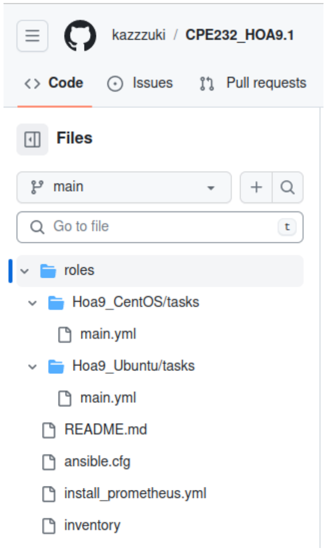
```
[kazuki@centos ~]$ prometheus --version
bash: /usr/local/bin/prometheus: No such file or directory
[kazuki@centos ~]$ prometheus --version
prometheus, version 2.31.2 (branch: HEAD, revision: bd21aafb66d6f5cdd80466f72590816e2e3
9fc14)
  build user:       root@50e835767f60
  build date:       20211209-12:15:08
  go version:       go1.17.3
  platform:         linux/amd64
```

I used the "prometheus –version" command to show that my playbook successfully installed prometheus for my Ubuntu and CentOS servers.

4. Make sure to create a new repository in GitHub for this activity.



```
kazuki@workstation:~/CPE232_HOA9.1$ git add .
kazuki@workstation:~/CPE232_HOA9.1$ git commit -m "hoa9"
[main c382f21] hoa9
 5 files changed, 63 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 install_prometheus.yml
 create mode 100644 inventory
 create mode 100644 roles/Hoa9_CentOS/tasks/main.yml
 create mode 100644 roles/Hoa9_Ubuntu/tasks/main.yml
kazuki@workstation:~/CPE232_HOA9.1$ git push
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 2 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (12/12), 1.26 KiB | 647.00 KiB/s, done.
Total 12 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:kazzzuki/CPE232_HOA9.1.git
   dd68e4e..c382f21  main -> main
```



```
roles
  Hoa9_CentOS/tasks
    main.yml
  Hoa9_Ubuntu/tasks
    main.yml
README.md
ansible.cfg
install_prometheus.yml
inventory
```

-> All files created in this activity are all successfully saved in my GitHub repository. I check everything, all of them are the same as my home repository directory. Here is the link of my GitHub Repository:

https://github.com/kazzzuki/CPE232_HOA9.1

Reflections:
Answer the following:
1. What are the benefits of having a performance monitoring tool?
   - The benefits of having performance monitoring tools are detecting collecting data when there is an error encountered, improved performance, enhanced user experience, and security. This helps us identify the problems/errors efficiently and saves us time. For example, in this activity, the “Prometheus”, it helps in detecting and resolving performance issues quickly.

Conclusions:

This activity enhances my knowledge about creating a playbook using roles. I encountered some problems/errors but it is easy to troubleshoot since some of the errors I encountered are the same errors I encountered in activity 8.