Name: Buduan, Christian Aaron C.	**Date Performed:** 09/30/24
Course/Section: CPE31S2	**Date Submitted:** 09/30/24
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st sem 2024-2025

Activity 6: Targeting Specific Nodes and Managing Services

1. **Objectives:**

1.1 Individualize hosts

1.2 Apply tags in selecting plays to run

1.3 Managing Services from remote servers using playbooks

2. **Discussion**:

In this activity, we try to individualize hosts. For example, we don't want apache on all our servers, or maybe only one of our servers is a web server, or maybe we have different servers like databases or file servers running different things on different categories of servers and that is what we are going to take a look at in this activity.

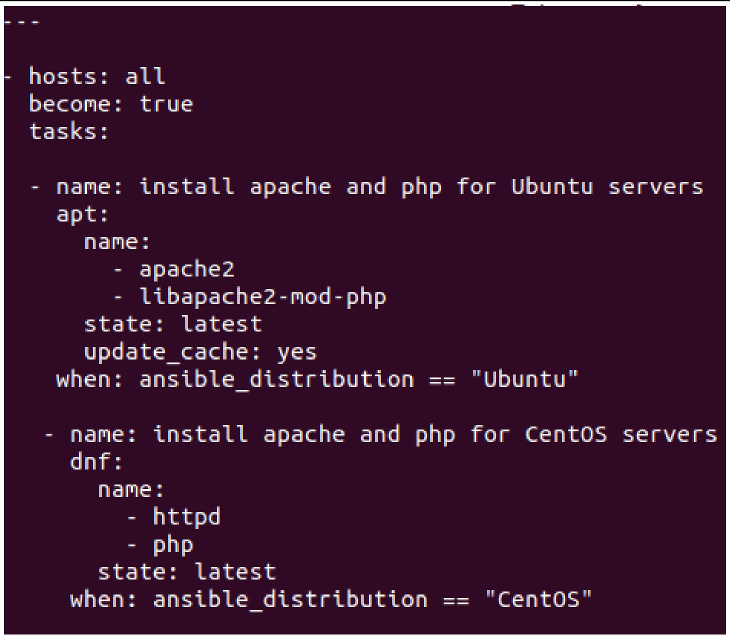
We also try to manage services that do not automatically run using the automations in the playbook. For example, when we install web servers or httpd for CentOS, we notice that the service did not start automatically.

Requirement:

In this activity, you will need to create another Ubuntu VM and name it Server 3. Likewise, you need to activate the second adapter to a host-only adapter after the installation. Take note of the IP address of Server 3. Make sure to use the command *ssh-copy-id* to copy the public key to Server 3. Verify if you can successfully SSH to Server 3.

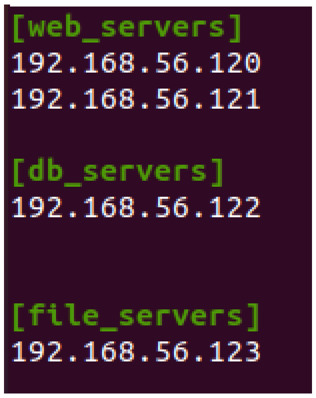
Task 1: Targeting Specific Nodes

1. Create a new playbook and name it site.yml. Follow the commands as shown in the image below. Make sure to save the file and exit.

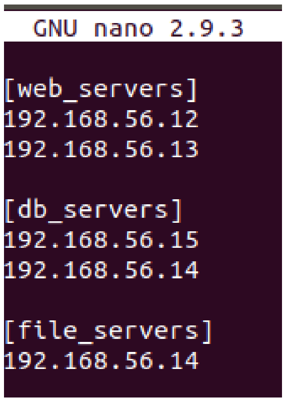


```
---

- hosts: all
  become: true
  tasks:

  - name: install apache and php for Ubuntu servers
    apt:
      name:
        - apache2
        - libapache2-mod-php
      state: latest
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

   - name: install apache and php for CentOS servers
     dnf:
       name:
         - httpd
         - php
       state: latest
     when: ansible_distribution == "CentOS"
```

2. Edit the inventory file. Remove the variables we put in our last activity and group according to the image shown below:



```
[web_servers]
192.168.56.120
192.168.56.121

[db_servers]
192.168.56.122


[file_servers]
192.168.56.123
```



```
  GNU nano 2.9.3

[web_servers]
192.168.56.12
192.168.56.13

[db_servers]
192.168.56.15
192.168.56.14

[file_servers]
192.168.56.14
```

192.168.56.12 = server1

192.168.56.13 = server2

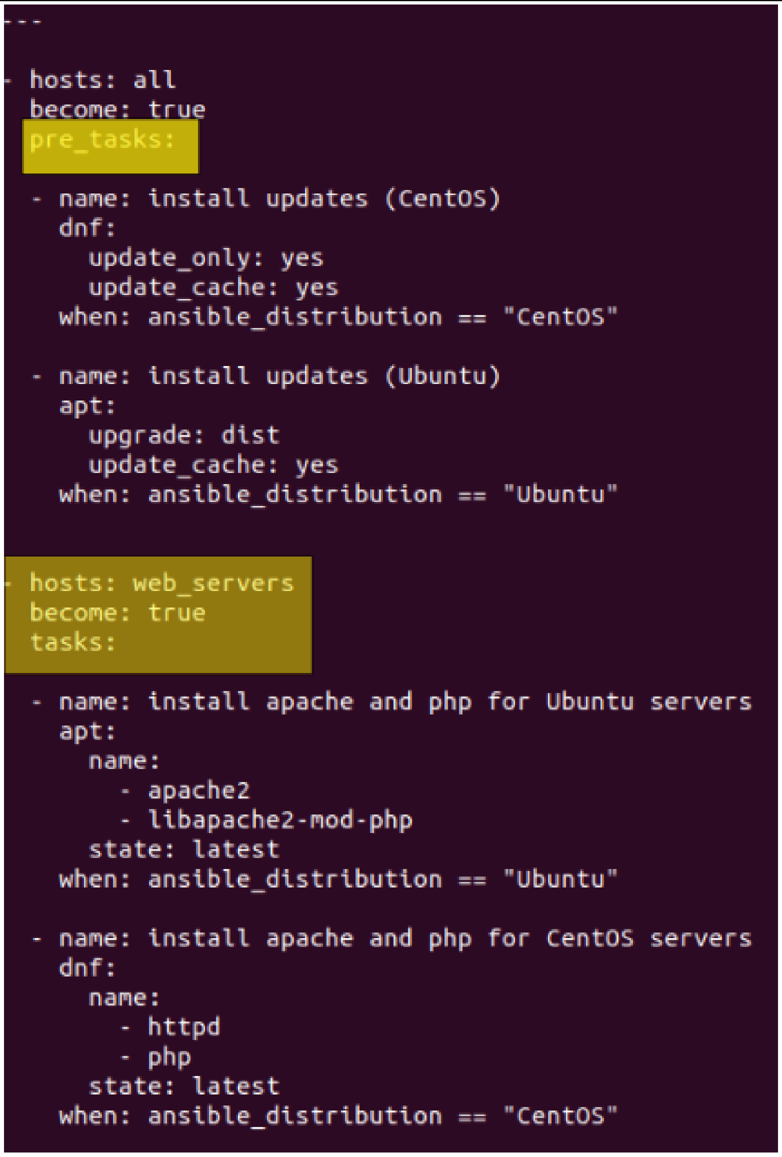
192.168.56.14 = centos server

192.168.56.15 = server3

Make sure to save the file and exit.

Right now, we have created groups in our inventory file and put each server in its own group. In other cases, you can have a server be a member of multiple groups, for example you have a test server that is also a web server.

3. Edit the *site.yml* by following the image below:

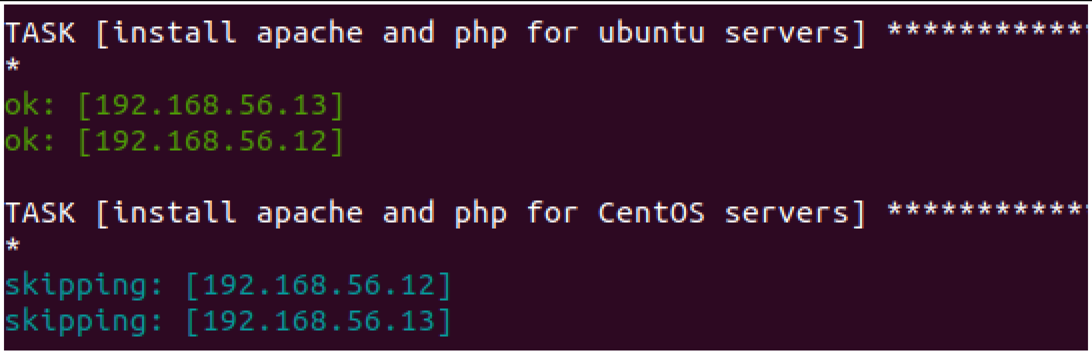


```
---

- hosts: all
  become: true
  pre_tasks:

  - name: install updates (CentOS)
    dnf:
      update_only: yes
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: install updates (Ubuntu)
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"


- hosts: web_servers
  become: true
  tasks:

  - name: install apache and php for Ubuntu servers
    apt:
      name:
        - apache2
        - libapache2-mod-php
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: install apache and php for CentOS servers
    dnf:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution == "CentOS"
```

Make sure to save the file and exit.

The *pre-tasks* command tells the ansible to run it before any other thing. In the *pre-tasks*, CentOS will install updates while Ubuntu will upgrade its distribution package. This will run before running the second play, which is targeted at *web_servers*. In the second play, apache and php will be installed on both Ubuntu servers and CentOS servers.

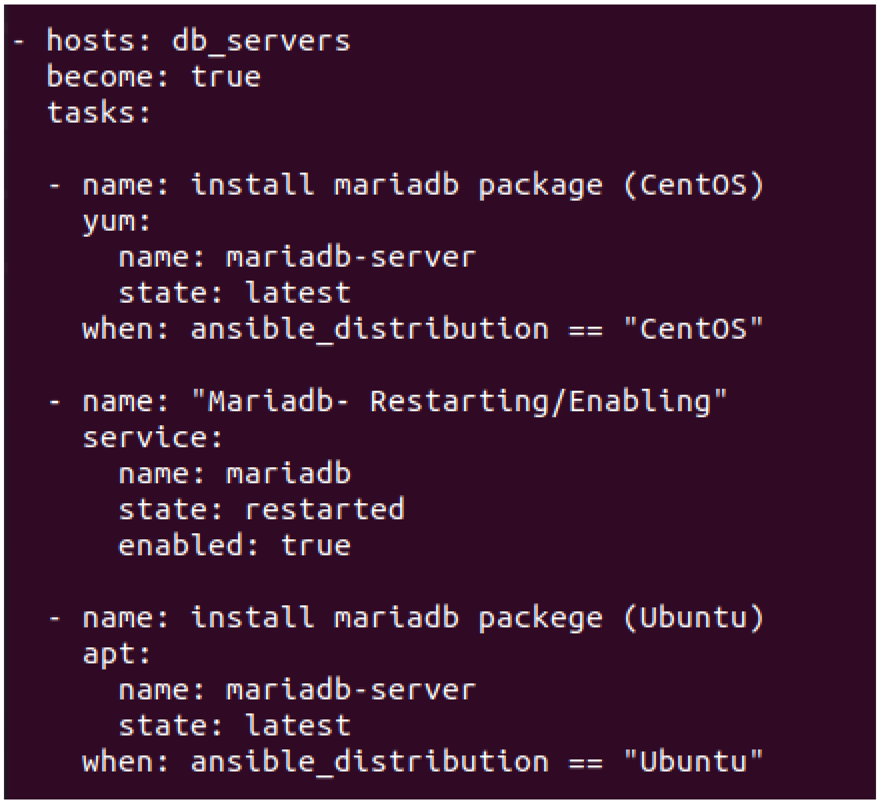
Run the *site.yml* file and describe the result.
After running the file, we notice that the tasks were only done in the ip addresses included in the web_server under the inventory file.



```
TASK [install apache and php for ubuntu servers] ************
*
ok: [192.168.56.13]
ok: [192.168.56.12]

TASK [install apache and php for CentOS servers] ************
*
skipping: [192.168.56.12]
skipping: [192.168.56.13]
```

play recap

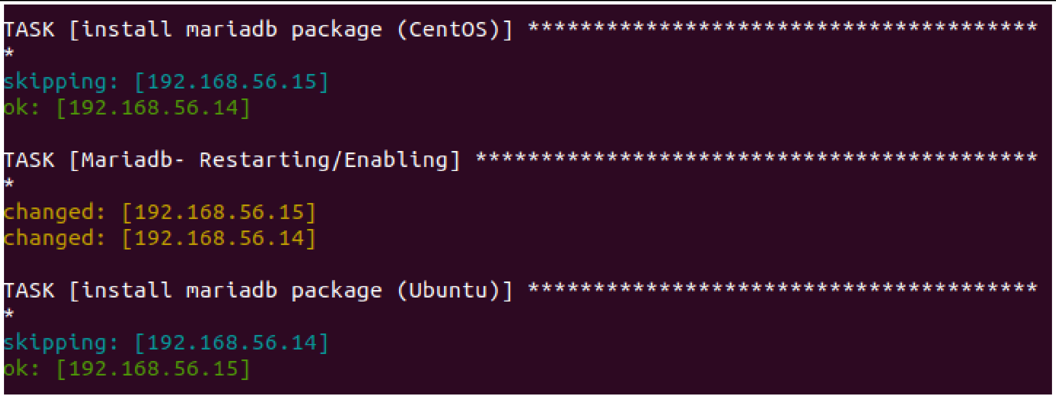
4. Let's try to edit again the *site.yml* file. This time, we are going to add plays targeting the other servers. This time we target the *db_servers* by adding it on the current *site.yml*. Below is an example: (Note add this at the end of the playbooks from task 1.3.



```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    yum:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true

  - name: install mariadb packege (Ubuntu)
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"
```

Make sure to save the file and exit.

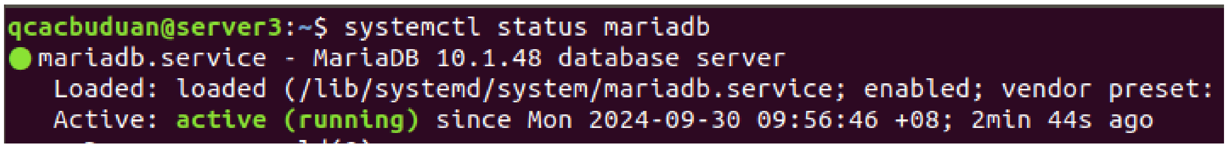
Run the *site.yml* file and describe the result.
After running the file, we can notice that the new tasks that were added only affected ip addresses that are under the db_servers.



```
TASK [install mariadb package (CentOS)] ****************************************
*
skipping: [192.168.56.15]
ok: [192.168.56.14]

TASK [Mariadb- Restarting/Enabling] ********************************************
*
changed: [192.168.56.15]
changed: [192.168.56.14]

TASK [install mariadb package (Ubuntu)] ****************************************
*
skipping: [192.168.56.14]
ok: [192.168.56.15]
```

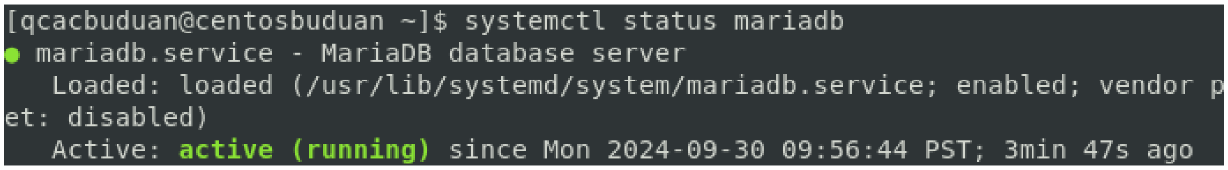
5. Go to the remote server (Ubuntu) terminal that belongs to the db_servers group and check the status for mariadb installation using the command: *systemctl status mariadb.* Do this on the CentOS server also.

   Describe the output. we can notice that after running the mariadb task, the status of mariadb is now active.



```
qcacbuduan@server3:~$ systemctl status mariadb
● mariadb.service - MariaDB 10.1.48 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset:
   Active: active (running) since Mon 2024-09-30 09:56:46 +08; 2min 44s ago
```

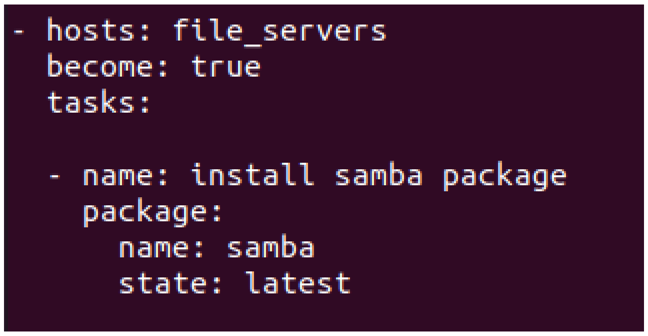
verifying mariadb status on db_server 192.168.56.15



```
[qcacbuduan@centosbuduan ~]$ systemctl status mariadb
● mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor p
et: disabled)
   Active: active (running) since Mon 2024-09-30 09:56:44 PST; 3min 47s ago
```

verifying mariadb status on db_server 192.168.56.14

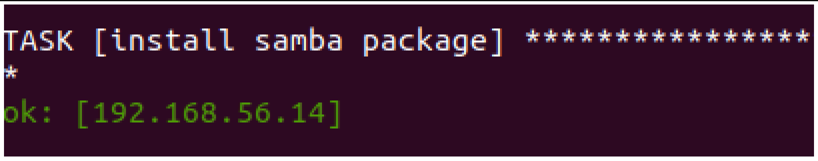
6. Edit the *site.yml* again. This time we will append the code to configure installation on the *file_servers* group. We can add the following on our file.



```
- hosts: file_servers
  become: true
  tasks:

  - name: install samba package
    package:
      name: samba
      state: latest
```

   Make sure to save the file and exit.

   Run the *site.yml* file and describe the result.
   This time, we ran a task that is exclusive to the manage nodes under file_servers.



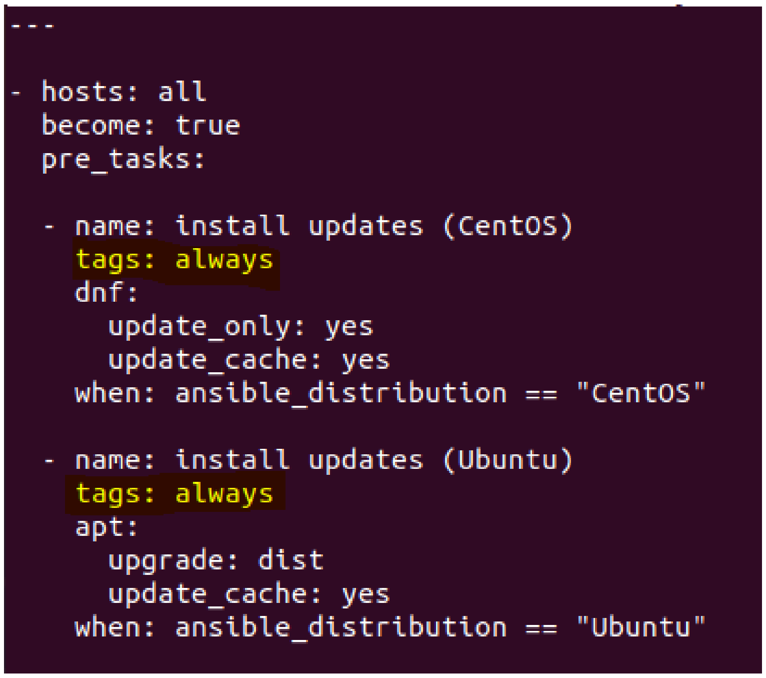
```
TASK [install samba package] *****************
*
ok: [192.168.56.14]
```

The testing of the *file_servers* is beyond the scope of this activity, and as well as our topics and objectives. However, in this activity we were able to show that we can target hosts or servers using grouping in ansible playbooks.

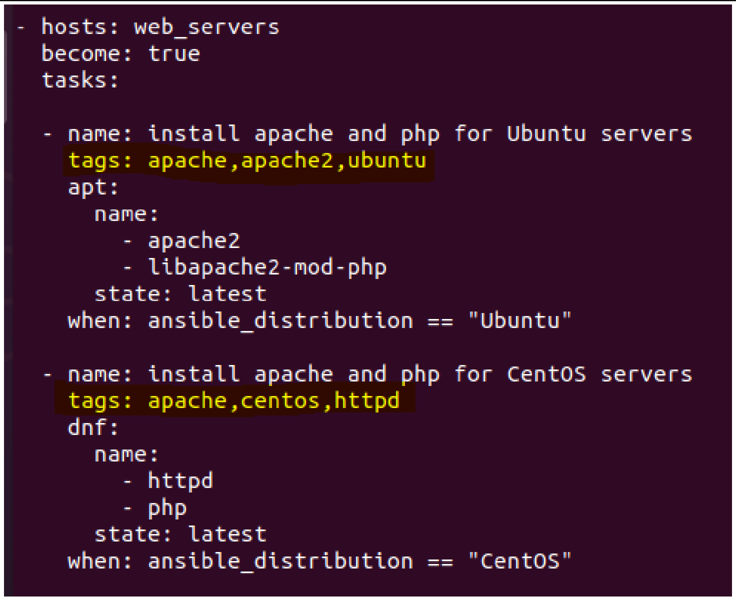
Task 2: Using Tags in running playbooks

In this task, our goal is to add metadata to our plays so that we can only run the plays that we want to run, and not all the plays in our playbook.

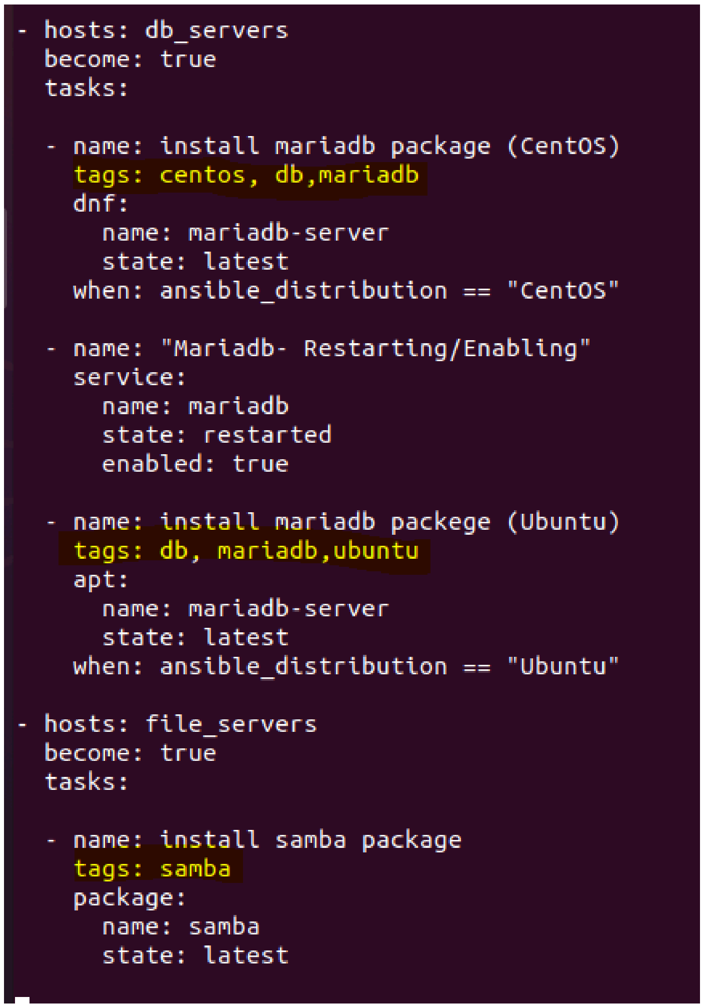
1. Edit the *site.yml* file. Add tags to the playbook. After the name, we can place the tags: *name_of_tag*. This is an arbitrary command, which means you can use any name for a tag.



```
---

- hosts: all
  become: true
  pre_tasks:

  - name: install updates (CentOS)
    tags: always
    dnf:
      update_only: yes
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: install updates (Ubuntu)
    tags: always
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"
```



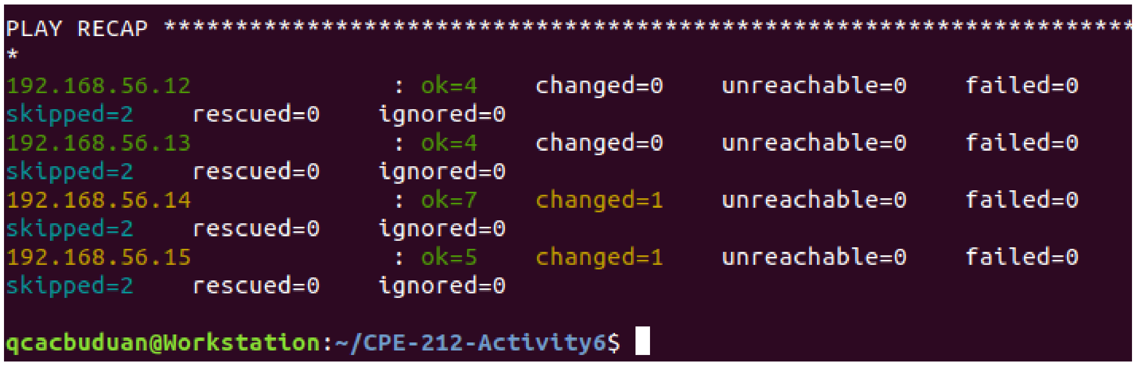
```
- hosts: web_servers
  become: true
  tasks:

  - name: install apache and php for Ubuntu servers
    tags: apache,apache2,ubuntu
    apt:
      name:
        - apache2
        - libapache2-mod-php
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: install apache and php for CentOS servers
    tags: apache,centos,httpd
    dnf:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution == "CentOS"
```



```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos, db,mariadb
    dnf:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true

  - name: install mariadb packege (Ubuntu)
    tags: db, mariadb,ubuntu
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"

- hosts: file_servers
  become: true
  tasks:

  - name: install samba package
    tags: samba
    package:
      name: samba
      state: latest
```

Make sure to save the file and exit.

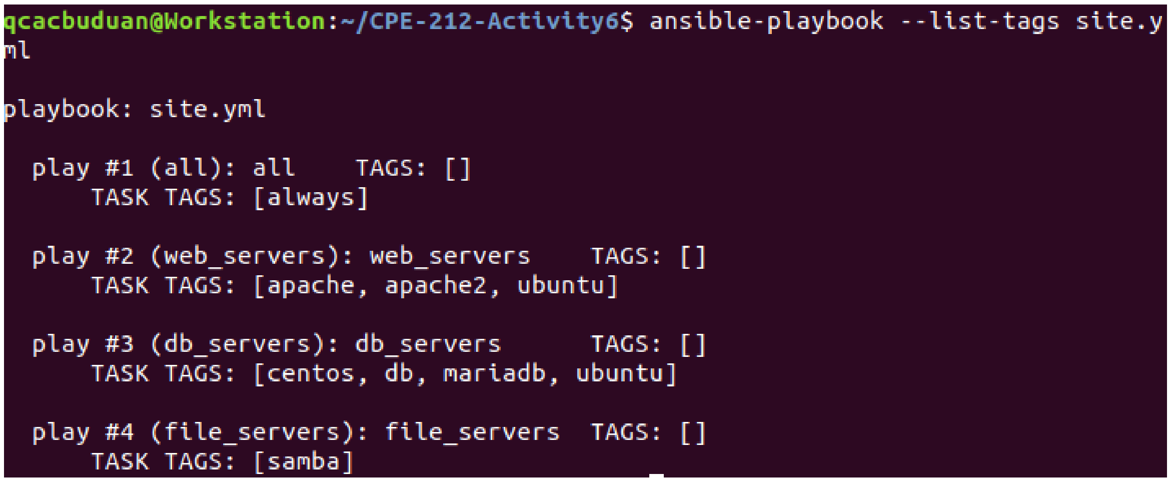
Run the *site.yml* file and describe the result. After putting the tags and run the playbook normally, there was no change that I have noticed.



```
PLAY RECAP *********************************************************************
*
192.168.56.12              : ok=4    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.13              : ok=4    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.14              : ok=7    changed=1    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.15              : ok=5    changed=1    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0

qcacbuduan@Workstation:~/CPE-212-Activity6$
```

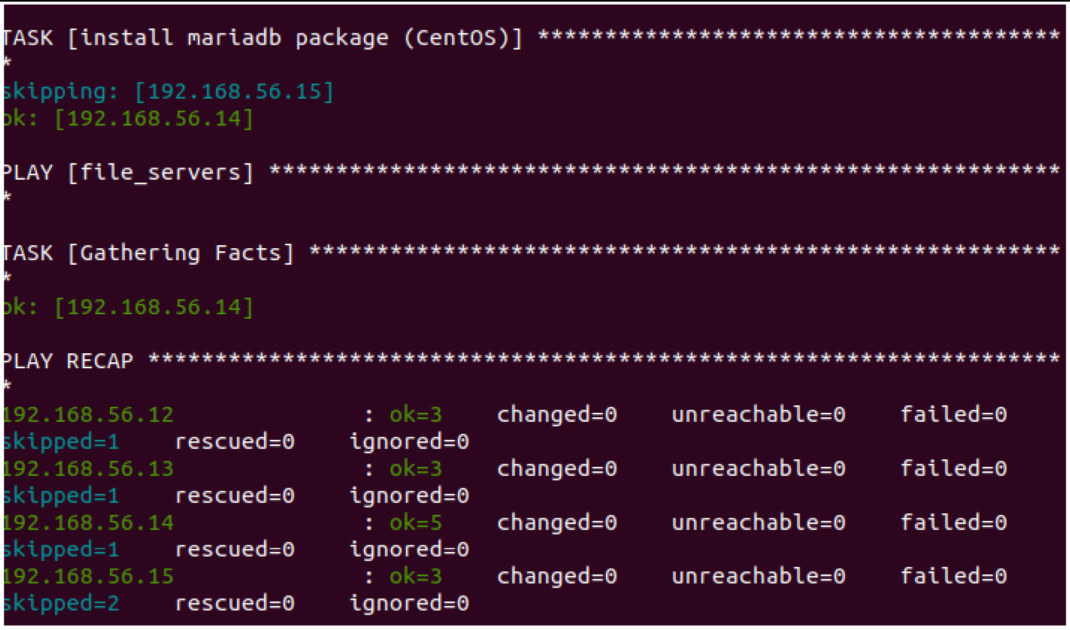
2. On the local machine, try to issue the following commands and describe each result:

   2.1 ansible-playbook --list-tags site.yml - using this commands shows us all the tags in the site.yml



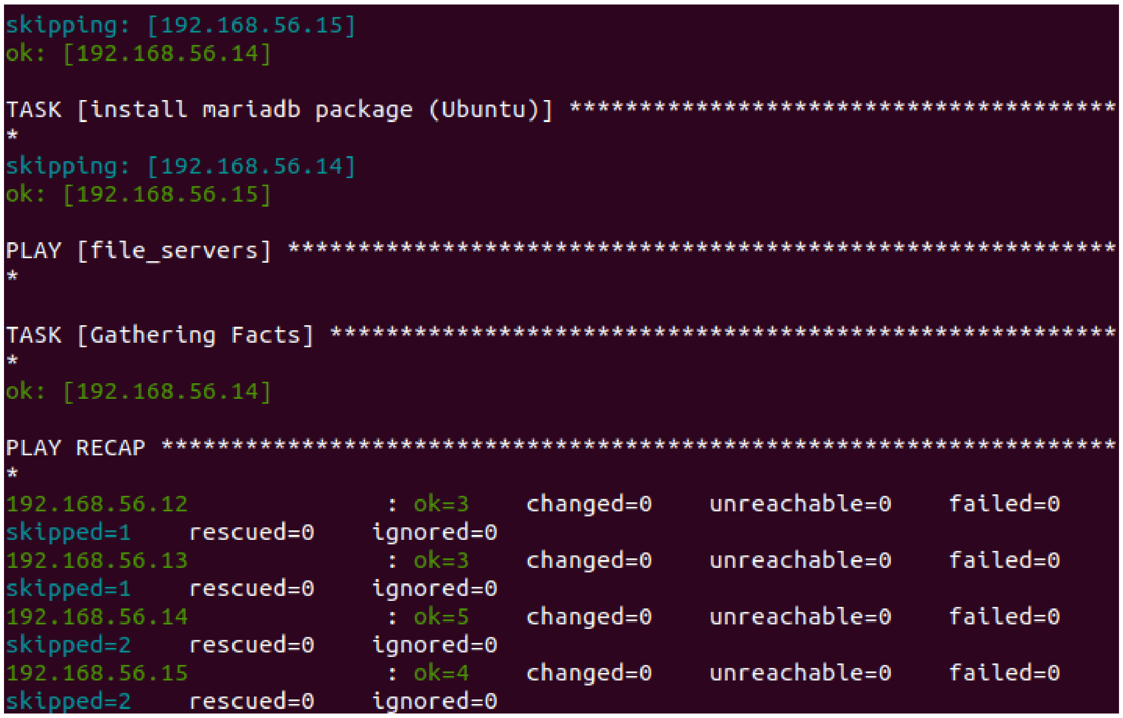
```
qcacbuduan@Workstation:~/CPE-212-Activity6$ ansible-playbook --list-tags site.y
ml

playbook: site.yml

  play #1 (all): all    TAGS: []
      TASK TAGS: [always]

  play #2 (web_servers): web_servers    TAGS: []
      TASK TAGS: [apache, apache2, ubuntu]

  play #3 (db_servers): db_servers      TAGS: []
      TASK TAGS: [centos, db, mariadb, ubuntu]

  play #4 (file_servers): file_servers  TAGS: []
      TASK TAGS: [samba]
```

   2.2 ansible-playbook --tags centos --ask-become-pass site.yml - this playbook runs with the tag centos which only does tasks on the centos server. you can notice that centos server has done the most task.



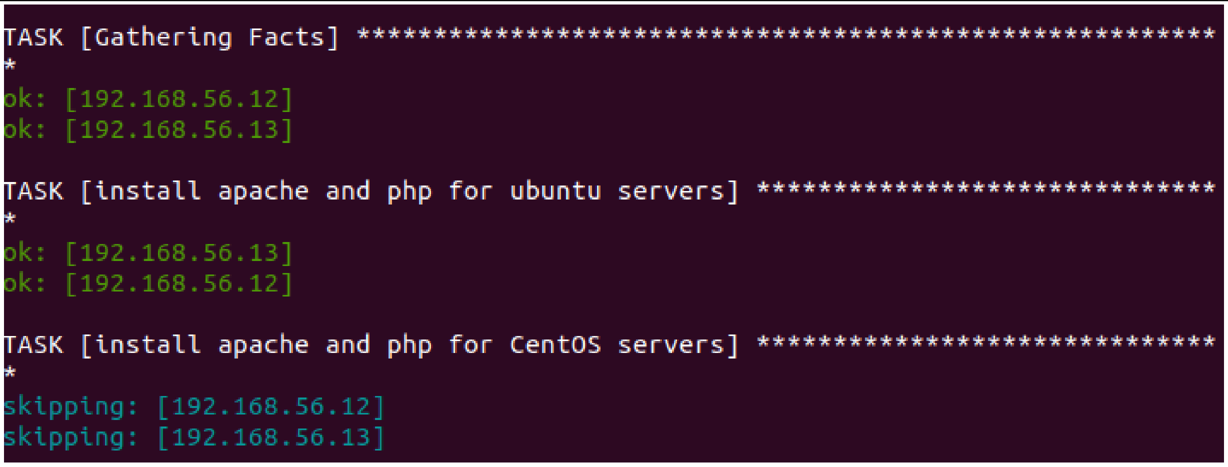
```
TASK [install mariadb package (CentOS)] *****************************************
*
skipping: [192.168.56.15]
ok: [192.168.56.14]

PLAY [file_servers] *************************************************************
*

TASK [Gathering Facts] **********************************************************
*
ok: [192.168.56.14]

PLAY RECAP **********************************************************************
*
192.168.56.12              : ok=3    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
192.168.56.13              : ok=3    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
192.168.56.14              : ok=5    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
192.168.56.15              : ok=3    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
```

2.3 *ansible-playbook --tags db --ask-become-pass site.yml -* in this playbook, it plays tasks on db_servers only.

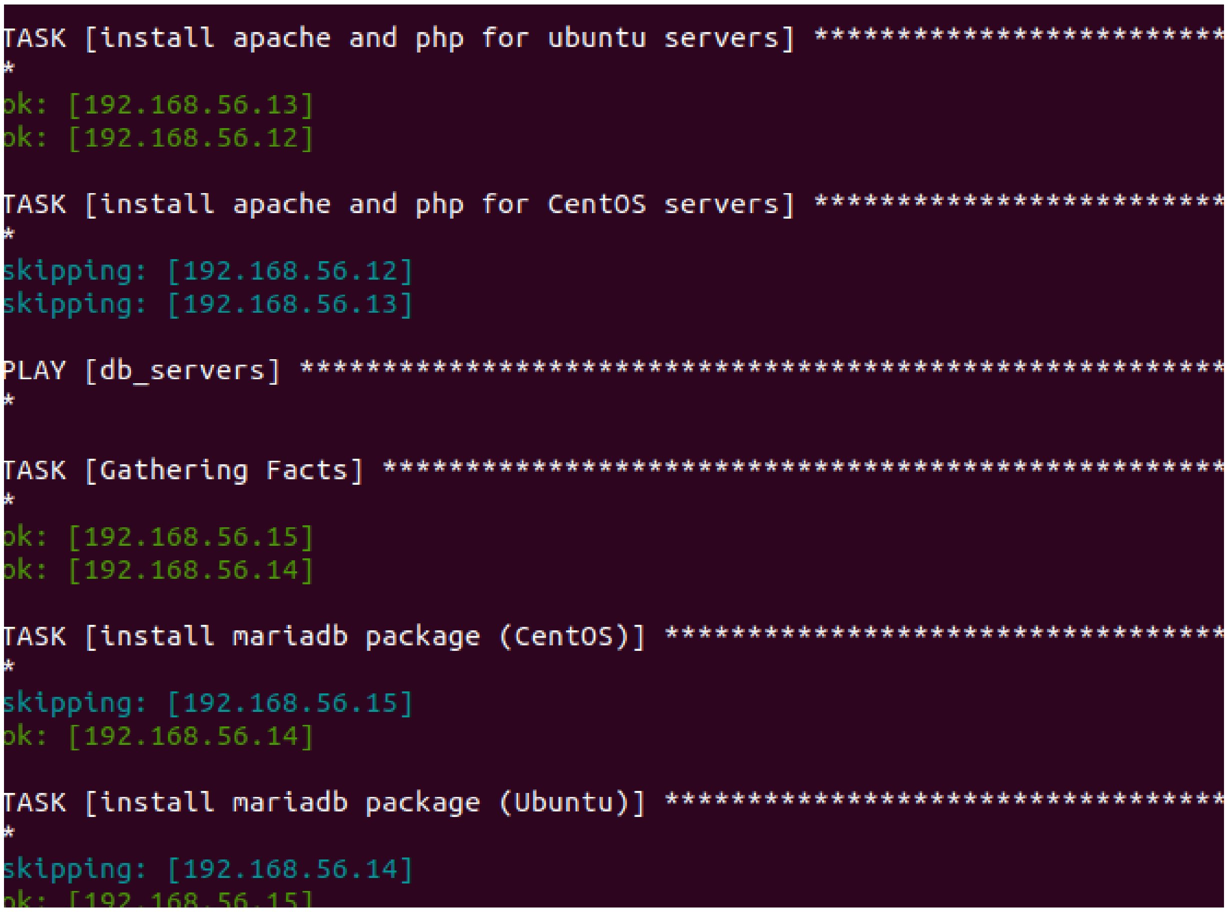


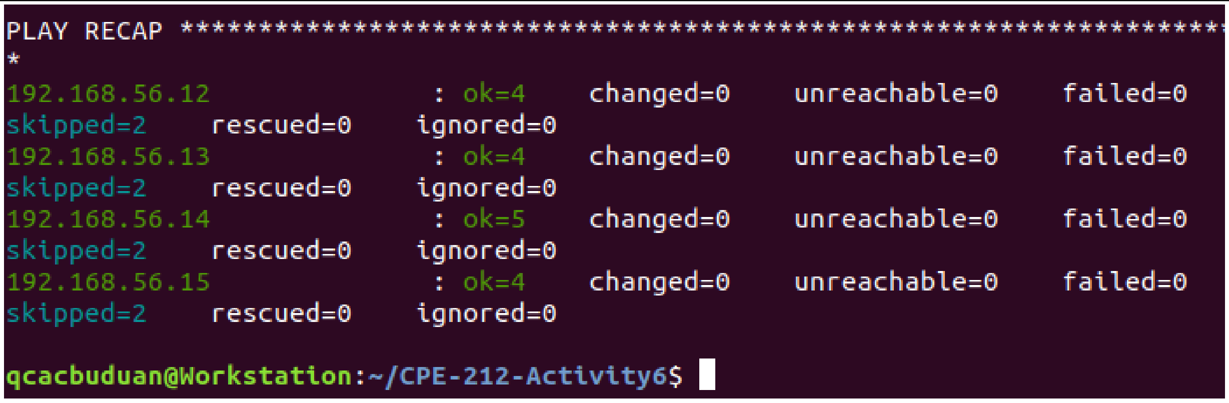
```
skipping: [192.168.56.15]
ok: [192.168.56.14]

TASK [install mariadb package (Ubuntu)] *****************************************
*
skipping: [192.168.56.14]
ok: [192.168.56.15]

PLAY [file_servers] *************************************************************
*

TASK [Gathering Facts] **********************************************************
*
ok: [192.168.56.14]

PLAY RECAP **********************************************************************
*
192.168.56.12              : ok=3    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
192.168.56.13              : ok=3    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
192.168.56.14              : ok=5    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.15              : ok=4    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
```

2.4 *ansible-playbook --tags apache --ask-become-pass site.yml -* in this play, it will only run tasks that has the tag “apache” which is only specified for web_servers.


```
TASK [Gathering Facts] ********************************************************
*
ok: [192.168.56.12]
ok: [192.168.56.13]

TASK [install apache and php for ubuntu servers] ******************************
*
ok: [192.168.56.13]
ok: [192.168.56.12]

TASK [install apache and php for CentOS servers] ******************************
*
skipping: [192.168.56.12]
skipping: [192.168.56.13]
```

2.5 *ansible-playbook --tags "apache,db" --ask-become-pass site.yml -* this command will make it so that only the tags with "db", and "apache" will only run its task.

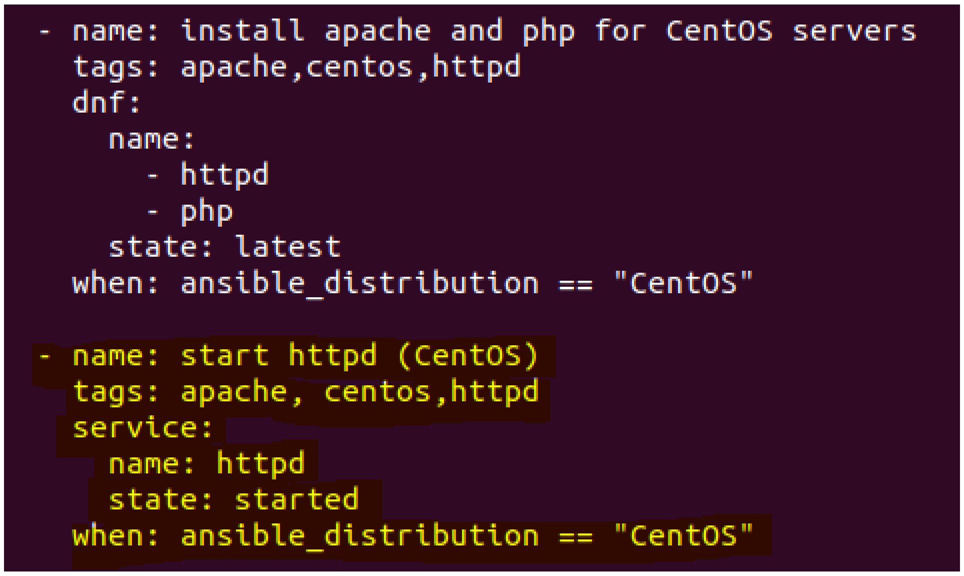

```
TASK [install apache and php for ubuntu servers] *************************
*
ok: [192.168.56.13]
ok: [192.168.56.12]

TASK [install apache and php for CentOS servers] *************************
*
skipping: [192.168.56.12]
skipping: [192.168.56.13]

PLAY [db_servers] ********************************************************
*

TASK [Gathering Facts] ***************************************************
*
ok: [192.168.56.15]
ok: [192.168.56.14]

TASK [install mariadb package (CentOS)] **********************************
*
skipping: [192.168.56.15]
ok: [192.168.56.14]

TASK [install mariadb package (Ubuntu)] **********************************
*
skipping: [192.168.56.14]
ok: [192.168.56.15]
```



```
PLAY RECAP *********************************************************************
*
192.168.56.12              : ok=4    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.13              : ok=4    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.14              : ok=5    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.15              : ok=4    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0

qcacbuduan@Workstation:~/CPE-212-Activity6$
```

Task 3: Managing Services

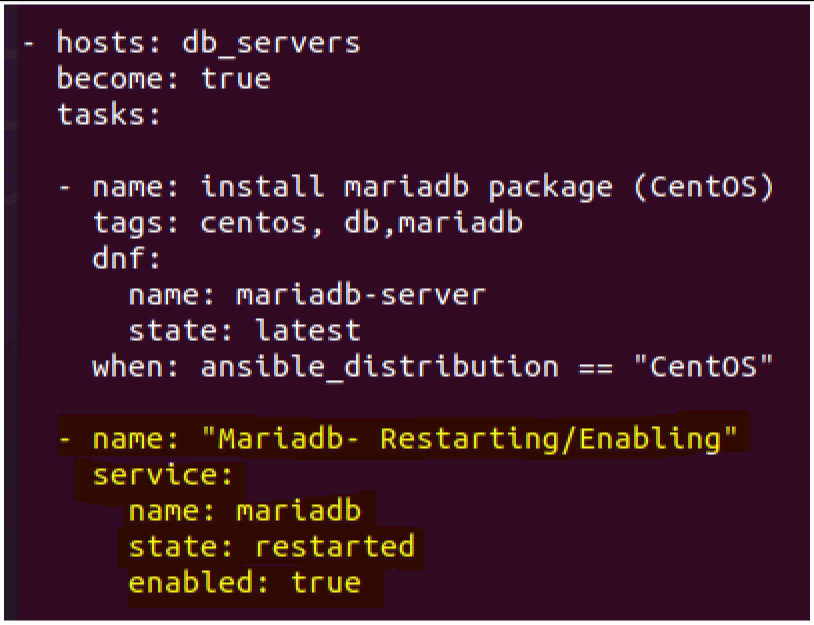
1. Edit the file site.yml and add a play that will automatically start the httpd on CentOS server.



```
- name: install apache and php for CentOS servers
  tags: apache,centos,httpd
  dnf:
    name:
      - httpd
      - php
    state: latest
  when: ansible_distribution == "CentOS"

- name: start httpd (CentOS)
  tags: apache, centos,httpd
  service:
    name: httpd
    state: started
  when: ansible_distribution == "CentOS"
```

Figure 3.1.1
Make sure to save the file and exit.

You would also notice from our previous activity that we already created a module that runs a service.

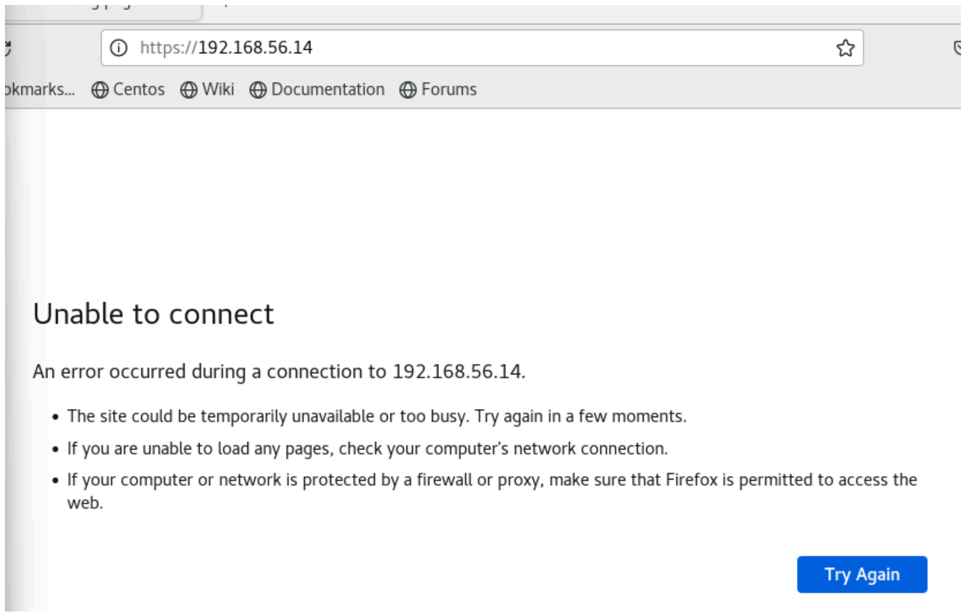


```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos, db,mariadb
    dnf:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true
```

Figure 3.1.2

This is because in CentOS, installed packages' services are not run automatically. Thus, we need to create the module to run it automatically.

2. To test it, before you run the saved playbook, go to the CentOS server and stop the currently running httpd using the command *sudo systemctl stop httpd.* When prompted, enter the sudo password. After that, open the browser and enter the CentOS server's IP address. You should not be getting a display because we stopped the httpd service already.



Unable to connect

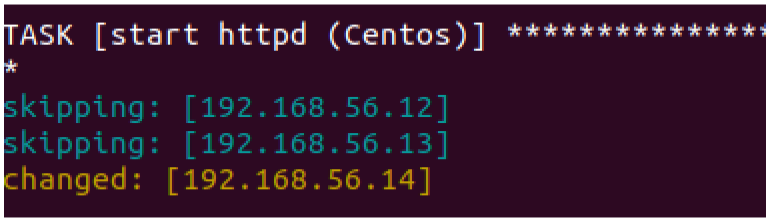
An error occurred during a connection to 192.168.56.14.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the web.

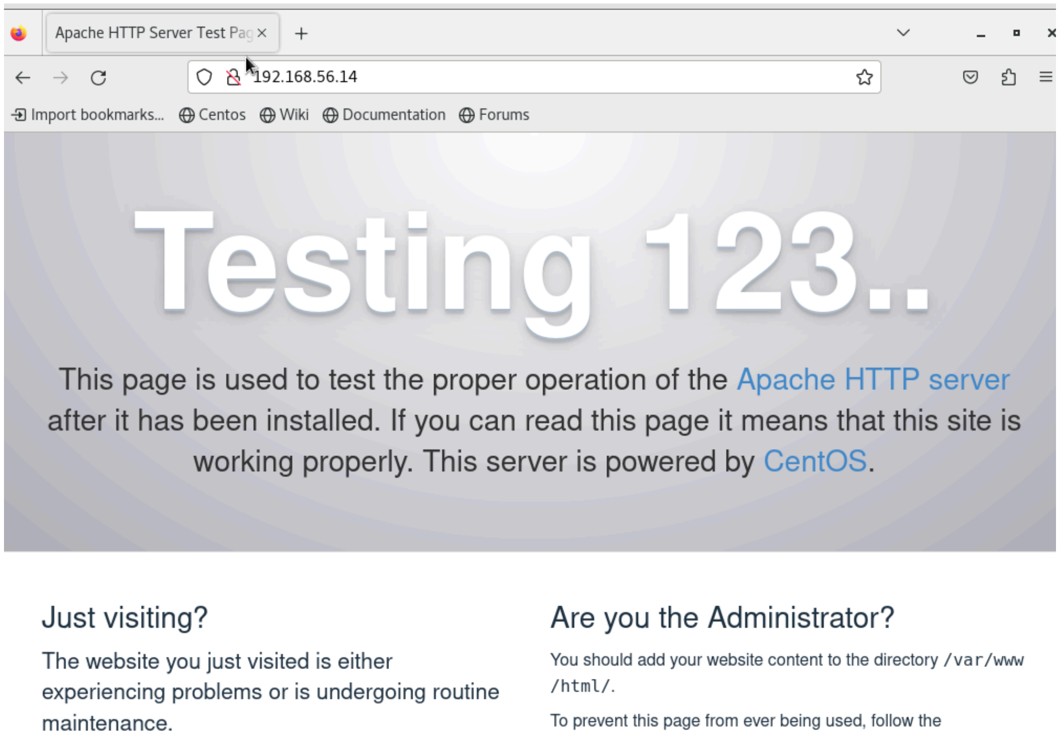
3. Go to the local machine and this time, run the *site.yml* file. Then after running the file, go again to the CentOS server and enter its IP address on the browser.

Describe the result. After running the file, the centos node was able to restart the "httpd" as shown in the results below.

note: before running, I added the centos' ip address under the web_server in my inventory file.



```
TASK [start httpd (Centos)] ****************
*
skipping: [192.168.56.12]
skipping: [192.168.56.13]
changed: [192.168.56.14]
```



Testing 123..

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page it means that this site is working properly. This server is powered by CentOS.

Just visiting?

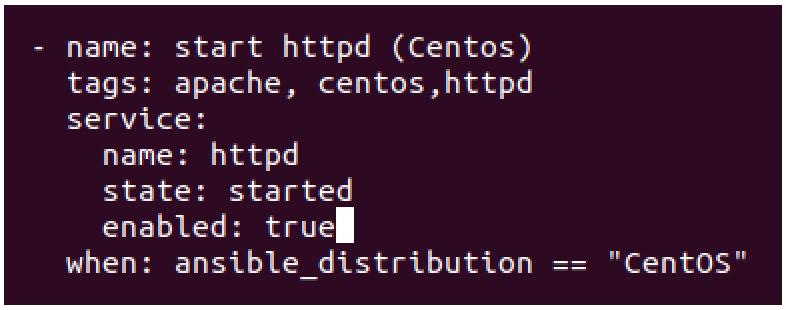
The website you just visited is either experiencing problems or is undergoing routine maintenance.

Are you the Administrator?

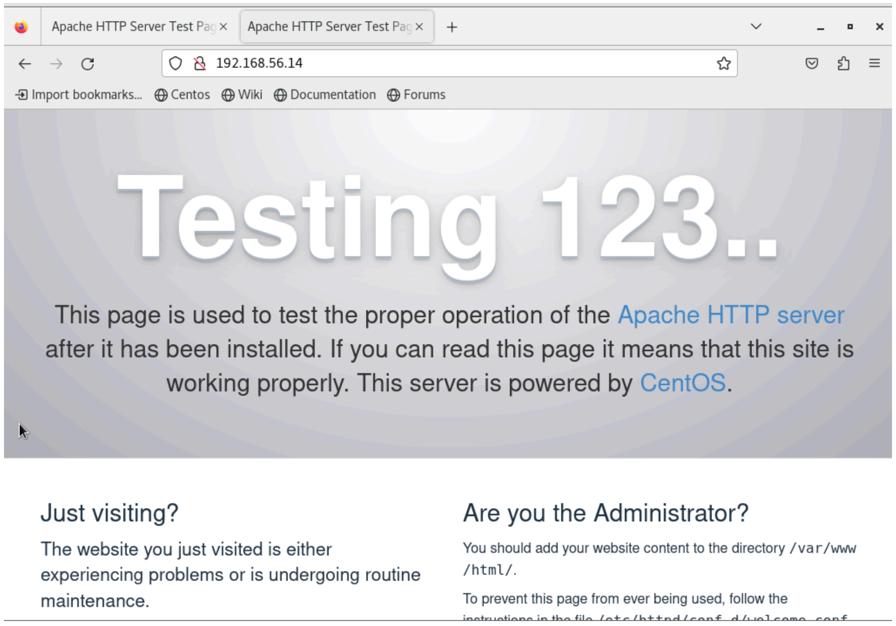
You should add your website content to the directory /var/www/html/.

To prevent this page from ever being used, follow the

To automatically enable the service every time we run the playbook, use the command *enabled: true* similar to Figure 7.1.2 and save the playbook.



```
- name: start httpd (Centos)
  tags: apache, centos,httpd
  service:
    name: httpd
    state: started
    enabled: true
  when: ansible_distribution == "CentOS"
```



Testing 123..

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page it means that this site is working properly. This server is powered by CentOS.

Just visiting?

The website you just visited is either experiencing problems or is undergoing routine maintenance.

Are you the Administrator?

You should add your website content to the directory `/var/www/html/`.

To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`

Github Repo: https://github.com/buduman/CPE-212-Activity6.git

Reflections:

Answer the following:

1. **What is the importance of putting our remote servers into groups?**
   The importance of distributing remote servers into groups is that we can assign and run tasks on the desired remote servers instead of having to specify them one by one in the playbook, making your playbook shorter, and organized.
2. **What is the importance of tags in playbooks?**
   The importance of tags in a playbook is the same as putting remote servers into groups, however, the difference is that we can assign which specific task will run, also making the playbook organized.
3. **Why do you think some services need to be managed automatically in playbooks?**

   Some of the services need to be managed automatically because some of these services are inactive when we turn on the remote servers, this relieves us of having to manually turn these services on each time.