

Name: Magboo, Matt Clemence C.	Date Performed: 09/12/2025
Course/Section: CPE31S2	Date Submitted: 09/12/2025
Instructor: Engr. Robin Valenzuela	Semester and SY: 2025-2026

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 database 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 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 installations. Take note of the IP address of the 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 named 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"
```

```
GNU nano 7.2                               Magboo@LocalMachine: ~/CPE212_Magboo
---                                         site.yml

- 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"

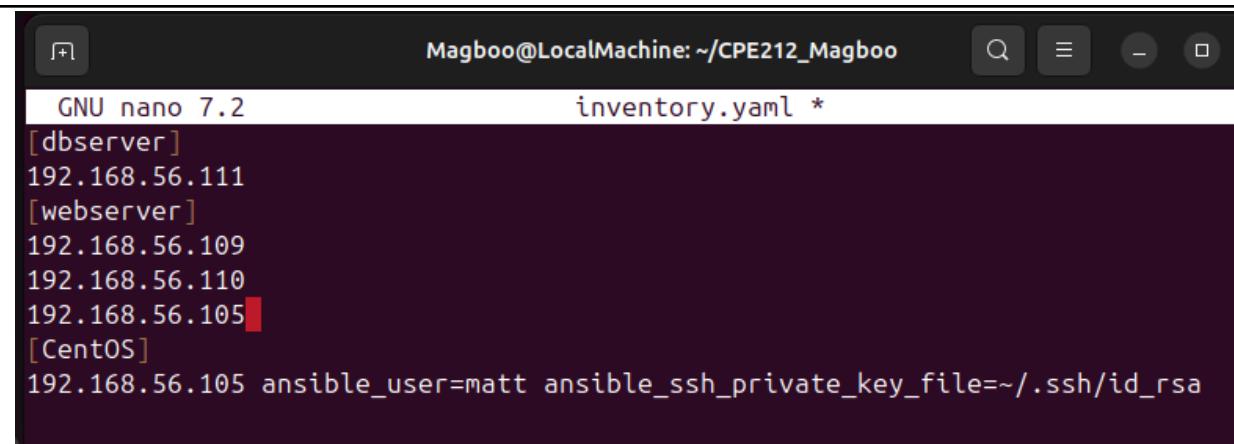
    - name: install apache and php for CentOS servers
      dnf:
        name:
          - httpd
          - php
        state: latest
      when: ansible_distribution == "CentOS"
```

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
```



The screenshot shows a terminal window titled "Magboo@LocalMachine: ~/CPE212_Magboo". The command "GNU nano 7.2" is displayed at the top. The file "inventory.yaml" is open, containing the following YAML configuration:

```
inventory.yaml *
[dbserver]
192.168.56.111
[webserver]
192.168.56.109
192.168.56.110
192.168.56.105
[CentOS]
192.168.56.105 ansible_user=matt ansible_ssh_private_key_file=~/ssh/id_rsa
```

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"
```

GNU nano 7.2

site.yml

```
---
```

- hosts: all
become: true
pre_tasks:
 - name: install updates (CentOS)
dnf:
 - name:
 - 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: webserver
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.

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.

```

Magboo@LocalMachine:~/CPE212_Magboo$ ansible-playbook --ask-become-pass site.yml
BECOME password:

PLAY [all] ****
TASK [Gathering Facts] ****
fatal: [192.168.56.111]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: no such identity: /home/Magboo/.ssh/ansible: No such file or directory\r\nMagboo@192.168.56.111: Permission denied (publickey,password).", "unreachable": true}
ok: [192.168.56.110]
ok: [192.168.56.109]
ok: [192.168.56.105]

TASK [install updates (CentOS)] ****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
fatal: [192.168.56.105]: FAILED! => {"changed": false, "msg": "argument 'name' is of type <class 'dict'> and we were unable to convert to list: <class 'dict'> cannot be converted to a list"}

TASK [install updates (Ubuntu)] ****
changed: [192.168.56.109]
changed: [192.168.56.110]

PLAY [webserver] ****
TASK [Gathering Facts] ****
ok: [192.168.56.110]
ok: [192.168.56.109]

TASK [install apache and php for ubuntu servers] ****
ok: [192.168.56.109]
ok: [192.168.56.110]

```

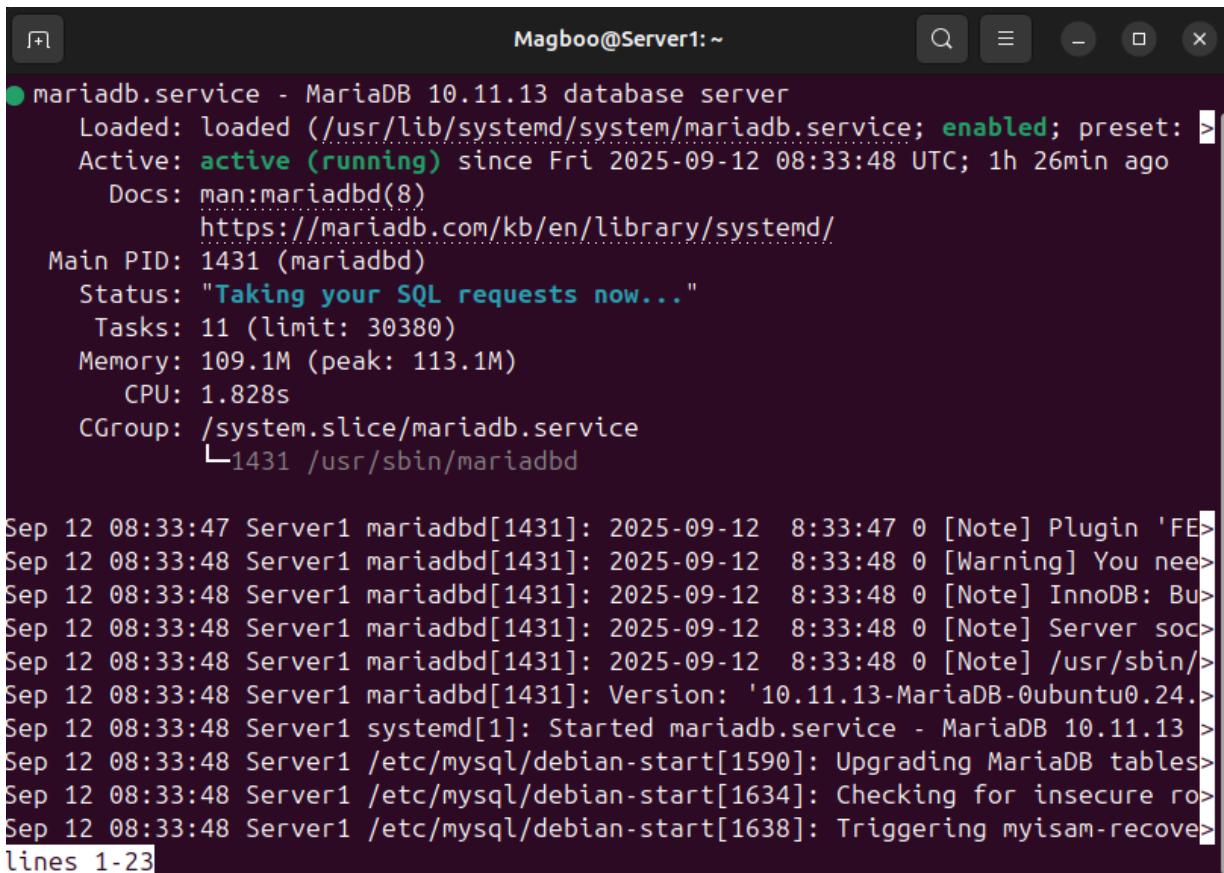
```
TASK [install apache and php for CentOS servers] ****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
```

```
PLAY RECAP ****
192.168.56.105 : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
192.168.56.109 : ok=4    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.110 : ok=4    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.111 : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
```

```
Magboo@LocalMachine:~/CPE212 Magboo$ sudo nano site.yml
```

it automatically skipped when the described OS is not what is needed when CentOS package is mentioned it will skip the ubuntu ip and vice versa.

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.



```
Magboo@Server1:~ mariadb.service - MariaDB 10.11.13 database server
  Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset: >
  Active: active (running) since Fri 2025-09-12 08:33:48 UTC; 1h 26min ago
    Docs: man:mariadb(8)
          https://mariadb.com/kb/en/library/systemd/
   Main PID: 1431 (mariadb)
     Status: "Taking your SQL requests now..."
      Tasks: 11 (limit: 30380)
     Memory: 109.1M (peak: 113.1M)
        CPU: 1.828s
       CGroup: /system.slice/mariadb.service
                 └─1431 /usr/sbin/mariadb

Sep 12 08:33:47 Server1 mariadb[1431]: 2025-09-12  8:33:47 0 [Note] Plugin 'FE>
Sep 12 08:33:48 Server1 mariadb[1431]: 2025-09-12  8:33:48 0 [Warning] You nee>
Sep 12 08:33:48 Server1 mariadb[1431]: 2025-09-12  8:33:48 0 [Note] InnoDB: Bu>
Sep 12 08:33:48 Server1 mariadb[1431]: 2025-09-12  8:33:48 0 [Note] Server soc>
Sep 12 08:33:48 Server1 mariadb[1431]: 2025-09-12  8:33:48 0 [Note] /usr/sbin/>
Sep 12 08:33:48 Server1 mariadb[1431]: Version: '10.11.13-MariaDB-0ubuntu0.24.>
Sep 12 08:33:48 Server1 systemd[1]: Started mariadb.service - MariaDB 10.11.13 >
Sep 12 08:33:48 Server1 /etc/mysql/debian-start[1590]: Upgrading MariaDB tables>
Sep 12 08:33:48 Server1 /etc/mysql/debian-start[1634]: Checking for insecure ro>
Sep 12 08:33:48 Server1 /etc/mysql/debian-start[1638]: Triggering myisam-recove>
lines 1-23
```

Describe the output.

mariadb is already enabled and running

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
```

```
- hosts: file_serves
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.

```
Sep 18 08:32
Magboo@LocalMachine: ~/CPE212_Magboo
PLAY [dbserver] ****
skipping: no hosts matched

PLAY [file_servers] ****

TASK [Gathering Facts] ****
ok: [192.168.56.105]
ok: [192.168.56.103]
ok: [192.168.56.104]

TASK [install samba package] ****
changed: [192.168.56.105]
changed: [192.168.56.103]
changed: [192.168.56.104]

PLAY RECAP ****
192.168.56.103      : ok=6    changed=1    unreachable=0    failed=0    s
kipped=2  rescued=0  ignored=0
192.168.56.104      : ok=6    changed=1    unreachable=0    failed=0    s
kipped=2  rescued=0  ignored=0
192.168.56.105      : ok=6    changed=1    unreachable=0    failed=0    s
kipped=2  rescued=0  ignored=0
```

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"
```

```
GNU nano 7.2  
---  
- hosts: all  
  become: true  
  pre_tasks:  
  
    - name: install updates (CentOS)  
      tags: always  
      dnf:  
        name:  
          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: webserver
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
```

```
- hosts: dbserver
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: insta;; mariadb package (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.

```

TASK [Mariadb- Restarting/Enabling] ****
fatal: [192.168.56.104]: FAILED! => {"changed": false, "msg": "Could not find the requested service mariadb: host"}
fatal: [192.168.56.103]: FAILED! => {"changed": false, "msg": "Could not find the requested service mariadb: host"}
changed: [192.168.56.105]

TASK [install mariadb package (Ubuntu)] ****
skipping: [192.168.56.105]

PLAY [file_servers] ****
TASK [Gathering Facts] ****
ok: [192.168.56.105]

TASK [install samba package] ****
ok: [192.168.56.105]

PLAY RECAP ****
192.168.56.103 : ok=5    changed=0    unreachable=0    failed=1    s
kiped=3    rescued=0    ignored=0
192.168.56.104 : ok=5    changed=0    unreachable=0    failed=1    s
kiped=3    rescued=0    ignored=0
192.168.56.105 : ok=9    changed=2    unreachable=0    failed=0    s
kiped=3    rescued=0    ignored=0

```

the tasked failed on mariadb restart because on ubuntu there is different command than centos

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

2.1 *ansible-playbook --list-tags site.yml*

```

Magboo@LocalMachine:~/CPE212_Magboo$ ansible-playbook --list-tags site.yml

playbook: site.yml

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

play #2 (webserver): webserver      TAGS: []
    TASK TAGS: [apache, apache2, centos, httpd, ubuntu]

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

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

```

it showed the task tags

2.2 *ansible-playbook --tags centos --ask-become-pass site.yml*

```
Sep 12 11:10
Magboo@LocalMachine: ~/CPE212_Magboo
PLAY [webserver] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [install apache and php for CentOS servers] *****
skipping: [192.168.56.109]
skipping: [192.168.56.110]

PLAY [dbserver] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [install mariadb package (CentOS)] *****
skipping: [192.168.56.109]
skipping: [192.168.56.110]

PLAY [file_servers] *****

PLAY RECAP *****
192.168.56.105      : ok=1    changed=0    unreachable=0    failed=1    skip
ped=0    rescued=0   ignored=0
192.168.56.109      : ok=4    changed=0    unreachable=0    failed=0    skip
ped=3    rescued=0   ignored=0
192.168.56.110      : ok=4    changed=0    unreachable=0    failed=0    skip
ped=3    rescued=0   ignored=0
```

it run everything but i still get the same error as last time

2.3 `ansible-playbook --tags db --ask-become-pass site.yml`

```
Magboo@LocalMachine:~/CPE212_Magboo$ ansible-playbook --tags db --ask-become-pass site.yml
BECOME password:

PLAY [all] ****
TASK [Gathering Facts] ****
ok: [192.168.56.105]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [install updates (CentOS)] ****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
fatal: [192.168.56.105]: FAILED! => {"changed": false, "msg": "argument 'name' is of type <class 'NoneType'> and we were unable to convert to list: <class 'NoneType'> cannot be converted to a list"}

TASK [install updates (Ubuntu)] ****
ok: [192.168.56.110]
ok: [192.168.56.109]

PLAY [webserver] ****
TASK [Gathering Facts] ****
ok: [192.168.56.109]
ok: [192.168.56.110]
```

it run the playbook that has tags of db

2.4 ansible-playbook --tags apache --ask-become-pass site.yml

```
Magboo@LocalMachine: ~/CPE212_Magboo
```

```
PLAY [webserver] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.109]

TASK [install apache and php for ubuntu servers] *****
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [install apache and php for CentOS servers] *****
skipping: [192.168.56.109]
skipping: [192.168.56.110]

PLAY [dbserver] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.109]

PLAY [file_servers] *****

PLAY RECAP *****
192.168.56.105          : ok=1    changed=0    unreachable=0    failed=1    skip
ped=0      rescued=0    ignored=0
192.168.56.109          : ok=5    changed=0    unreachable=0    failed=0    skip
ped=2      rescued=0    ignored=0
192.168.56.110          : ok=5    changed=0    unreachable=0    failed=0    skip
ped=2      rescued=0    ignored=0
```

it runs the playbook that has a tag of apache

2.5 `ansible-playbook --tags "apache,db" --ask-become-pass site.yml`

```
Magboo@LocalMachine:~/CPE212_Magboo$ ansible-playbook --tags "apache,db" --ask-become-pass site.yml
BECOME password:

PLAY [all] ****
TASK [Gathering Facts] ****
ok: [192.168.56.105]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [install updates (CentOS)] ****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
fatal: [192.168.56.105]: FAILED! => {"changed": false, "msg": "argument 'name' is of type <class 'NoneType'> and we were unable to convert to list: <class 'NoneType'> cannot be converted to a list"}

TASK [install updates (Ubuntu)] ****
ok: [192.168.56.109]
ok: [192.168.56.110]

PLAY [webserver] ****
TASK [Gathering Facts] ****
ok: [192.168.56.110]
ok: [192.168.56.109]

TASK [install apache and php for ubuntu servers] ****
```

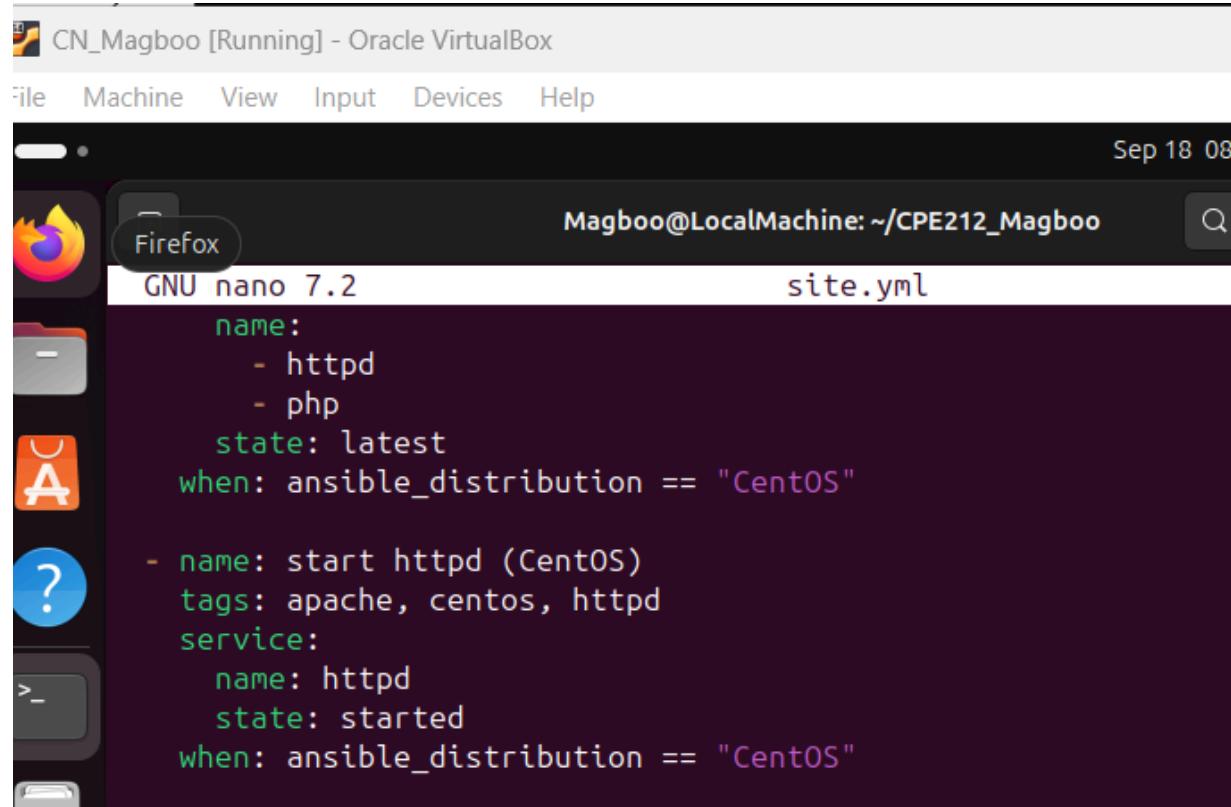
it run that has both the tags of apache and db

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"
```



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "Magboo@LocalMachine: ~/CPE212_Magboo". The terminal content displays the same Ansible YAML code as the previous block. The desktop interface includes a menu bar with "File", "Machine", "View", "Input", "Devices", and "Help". A dock on the left contains icons for "Firefox", "File Manager", "Terminal", "Apt", "Help", and "File Manager". The status bar at the bottom right shows the date and time as "Sep 18 08".

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

```

```

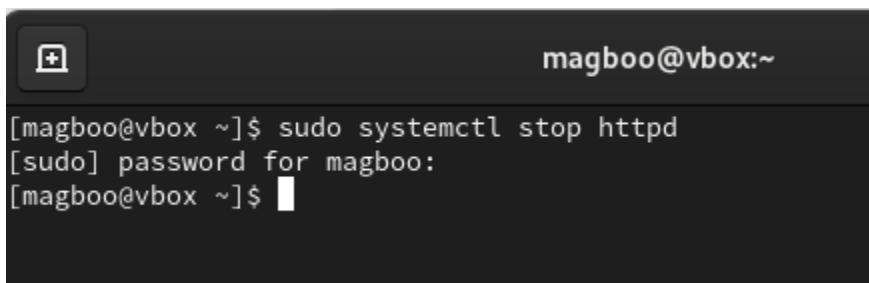
- name: "Mariadb- Restarting/Enabling"
service:
  name: mariadb
  state: restarted
  enabled: true

```

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.



The screenshot shows a terminal window with a dark background. At the top, there is a small icon with a plus sign. The prompt 'magboo@vbox:~' is visible. Below the prompt, the command '[magboo@vbox ~]\$ sudo systemctl stop httpd' is entered. A password prompt '[sudo] password for magboo:' follows, with a redacted password field. The command is completed with '[magboo@vbox ~]\$' at the bottom.

```

[magboo@vbox ~]$ sudo systemctl stop httpd
[sudo] password for magboo:
[magboo@vbox ~]$ 

```

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.

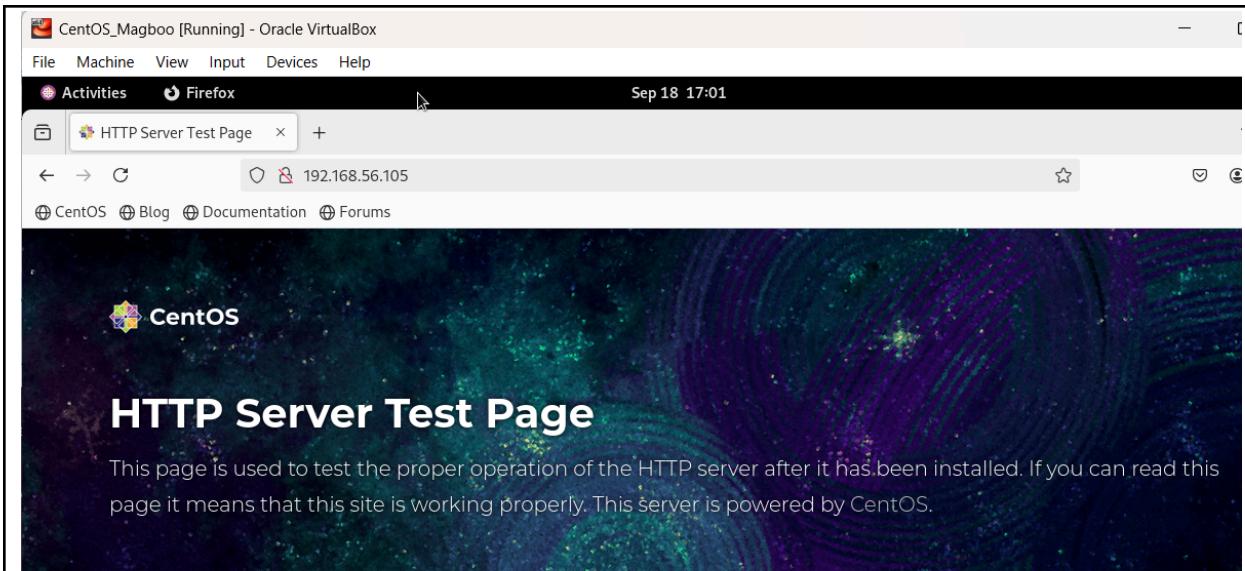
```
ok: [192.168.56.105]
TASK [Mariadb- Restarting/Enabling] *****
fatal: [192.168.56.104]: FAILED! => {"changed": false, "msg": "Could not find the requested service mariadb: host"}
fatal: [192.168.56.103]: FAILED! => {"changed": false, "msg": "Could not find the requested service mariadb: host"}
changed: [192.168.56.105]

TASK [install mariadb package (Ubuntu)] *****
skipping: [192.168.56.105]

PLAY [file_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.105]

TASK [install samba package] *****
ok: [192.168.56.105]

PLAY RECAP *****
192.168.56.103      : ok=5    changed=0    unreachable=0    failed=1    s
kipped=4  rescued=0  ignored=0
192.168.56.104      : ok=5    changed=0    unreachable=0    failed=1    s
kipped=4  rescued=0  ignored=0
192.168.56.105      : ok=10   changed=2    unreachable=0    failed=0    s
kipped=3  rescued=0  ignored=0
```



If you are a member of the general public:

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

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

If you are the website administrator:

You may now add content to the webroot directory. Note that until you do so, people visiting your website will see this page, and not your content.

For systems using the Apache HTTP Server:

You may now add content to the directory /var/www/html/. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from



though the command for the mariadb in centos and ubuntu are different since we called correctly for the mariadb in centos it still run ok though it failed to run for the ubuntu

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.

Reflections:

Answer the following:

1. What is the importance of putting our remote servers into groups?
so that you would know which should be installed in which server meaning you can pin point which computer that needs the update or install or which group they are

2. What is the importance of tags in playbooks?
It is important because it can be the control in which part of the playbook should be used instead of running everything in the playbook. You can pinpoint exactly what you

need and don't need to wait for it to run everything to know that the program runs properly.

3. Why do think some services need to be managed automatically in playbooks?

some services are too important to the systems that's why it needs to be automated like the firewall and the database specially for enterprises that depended on the network.