Name: Sembrero, Christian Dane D.	**Date Performed: 09/12/25**
Course/Section: CPE 212-CPE31S2	**Date Submitted: 08/12/25**
Instructor: Engr. Robin Valenzuela	**Semester and SY: 1st sem 2025-2026**

Activity 5: Consolidating Playbook plays

1. **Objectives:**

1.1 Use **when** command in playbook for different OS distributions

1.2 Apply refactoring techniques in cleaning up the playbook codes

2. **Discussion**:

We are going to look at a way that we can differentiate a playbook by a host in terms of which distribution the host is running. It's very common in most Linux shops to run multiple distributions, for example, Ubuntu shop or Debian shop and you need a different distribution for a one off-case or perhaps you want to run plays only on certain distributions.

It is a best practice in ansible when you are working in a collaborative environment to use the command git pull. git pull is a Git command used to update the local version of a repository from a remote. By default, git pull does two things. Updates the current local working branch (currently checked out branch) and updates the remote-tracking branches for all other branches. git pull essentially pulls down any changes that may have happened since the last time you worked on the repository.

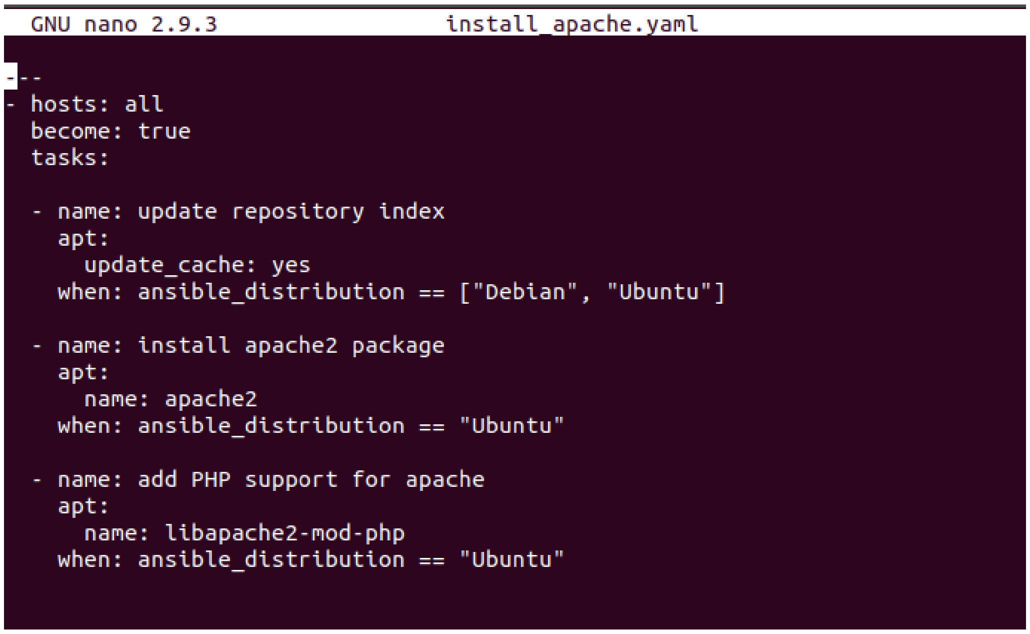
Requirement:
In this activity, you will need to create a CentOS VM. Likewise, you need to activate the second adapter to a host-only adapter after the installations. Take note of the IP address of the CentOS VM. Make sure to use the command *ssh-copy-id* to copy the public key to CentOS. Verify if you can successfully SSH to CentOS VM.

Task 1: Use when command for different distributions

1. In the local machine, make sure you are in the local repository directory (*CPE232_yourname*). Issue the command git pull. When prompted, enter the correct passphrase or password. Describe what happened when you issue this command. Did something happen? Why?
2. Edit the inventory file and add the IP address of the Centos VM. Issue the command we used to execute the playbook (the one we used in the last activity): *ansible-playbook --ask-become-pass install_apache.yml*. After executing this command, you may notice that it did not become successful in

the Centos VM. You can see that the Centos VM has failed=1. Only the two remote servers have been changed. The reason is that Centos VM does not support "apt" as the package manager. The default package manager for Centos is "yum."

3. Edit the *install_apache.yml* file and insert the lines shown below.



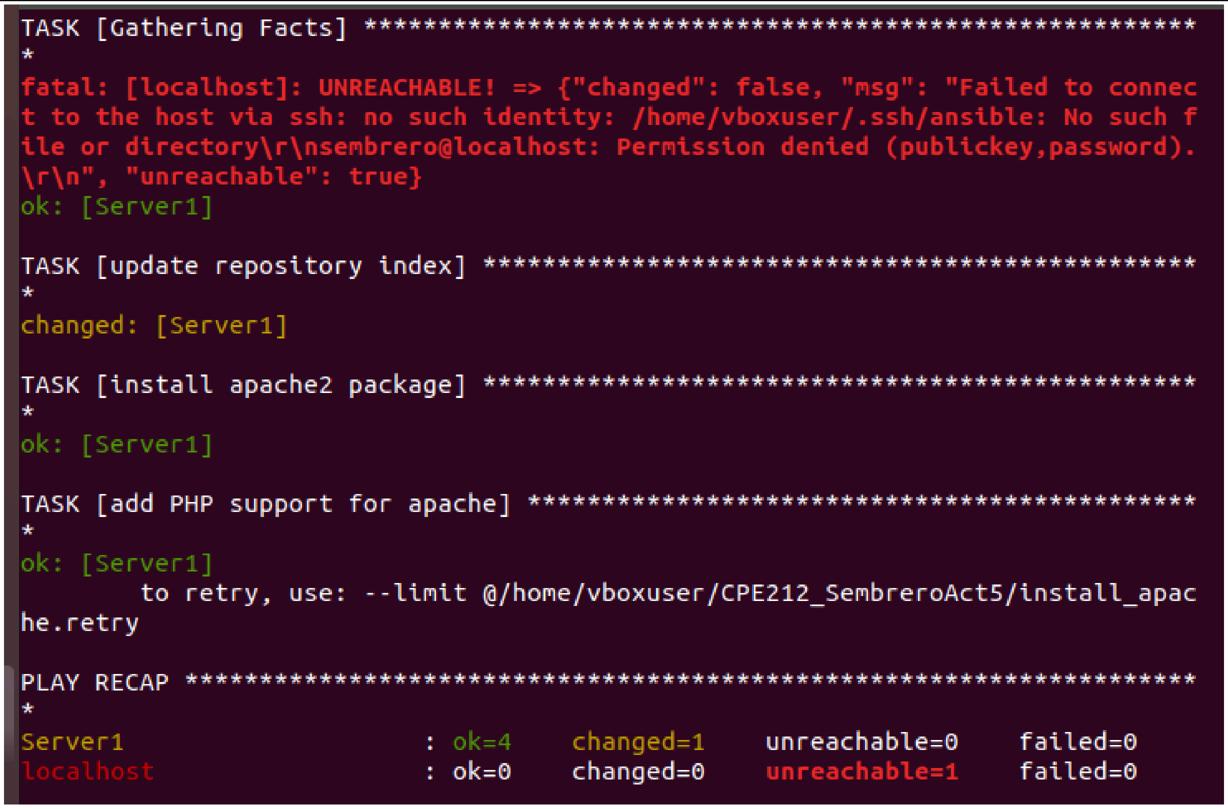
```
  GNU nano 2.9.3                    install_apache.yaml

---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution == ["Debian", "Ubuntu"]

  - name: install apache2 package
    apt:
      name: apache2
    when: ansible_distribution == "Ubuntu"

  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
    when: ansible_distribution == "Ubuntu"
```

Make sure to save the file and exit.

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.

If you have a mix of Debian and Ubuntu servers, you can change the configuration of your playbook like this.

```
-   name: update repository index
    apt:
       update_cache: yes
    when: ansible_distribution in [“Debian”, “Ubuntu]
```

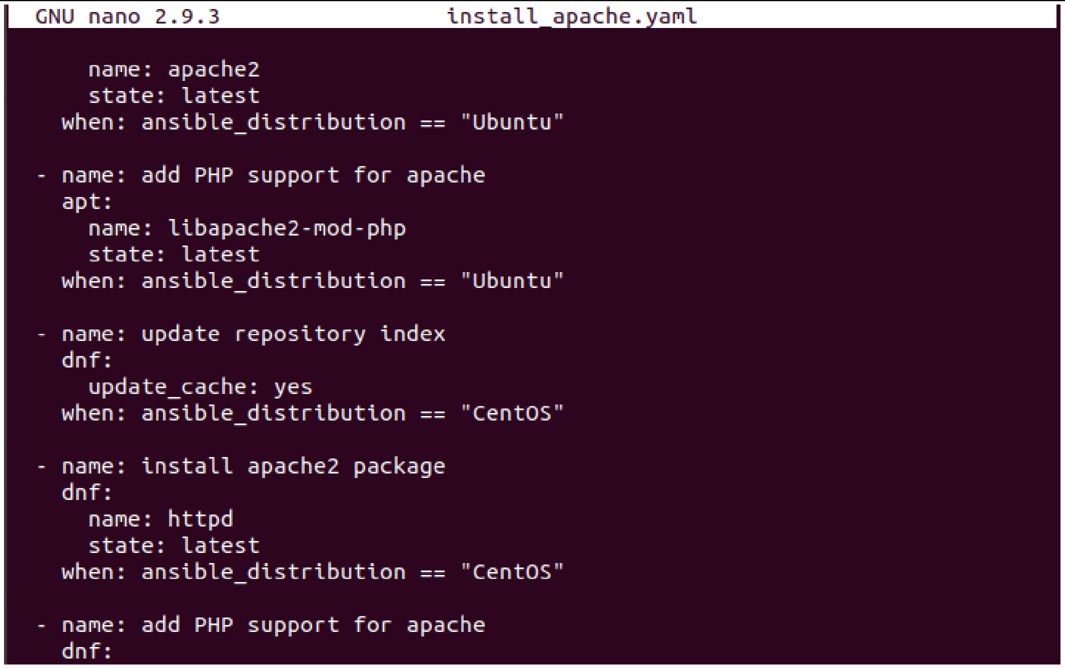


```
TASK [Gathering Facts] *********************************************************
*
fatal: [localhost]: UNREACHABLE! => {"changed": false, "msg": "Failed to connec
t to the host via ssh: no such identity: /home/vboxuser/.ssh/ansible: No such f
ile or directory\r\nsembrero@localhost: Permission denied (publickey,password).
\r\n", "unreachable": true}
ok: [Server1]

TASK [update repository index] *************************************************
*
changed: [Server1]

TASK [install apache2 package] *************************************************
*
ok: [Server1]

TASK [add PHP support for apache] **********************************************
*
ok: [Server1]
        to retry, use: --limit @/home/vboxuser/CPE212_SembreroAct5/install_apac
he.retry

PLAY RECAP *********************************************************************
*
Server1                    : ok=4    changed=1    unreachable=0    failed=0
localhost                  : ok=0    changed=0    unreachable=1    failed=0
```

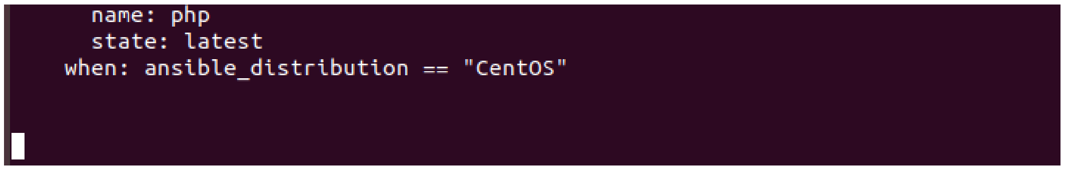
The ip 192.168.56.112 is from CentOS so some of commands inside playbook is not for CentOS, only for Ubuntu, which results in skipping CentOS from installing packages by playbook.

Note: This will work also if you try. Notice the changes are highlighted.

4. Edit the *install_apache.yml* file and insert the lines shown below.



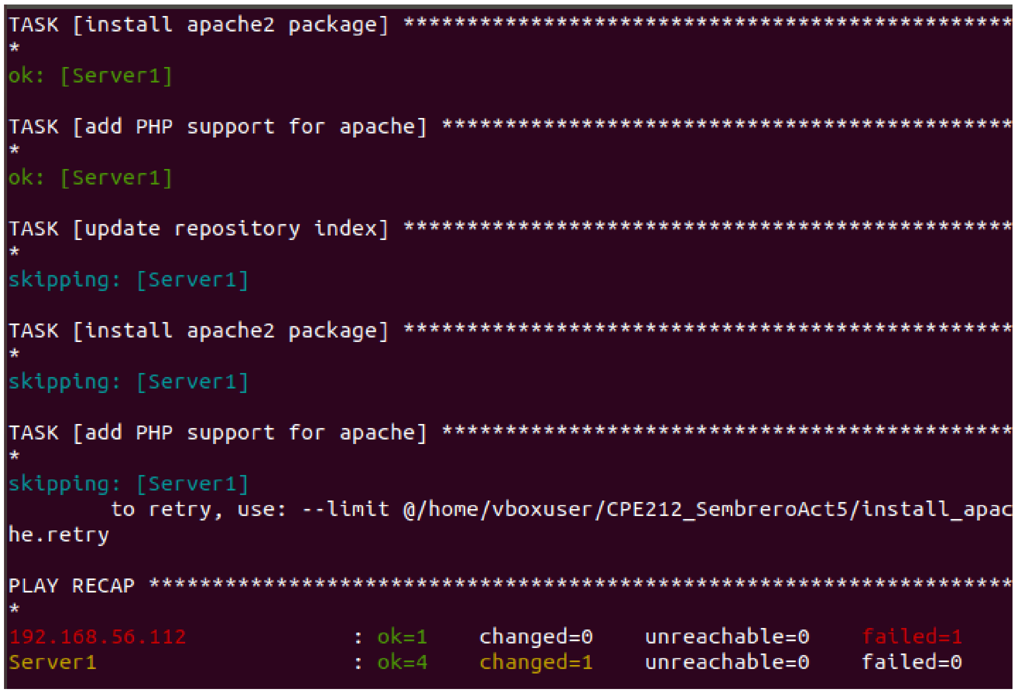
```
 GNU nano 2.9.3                install_apache.yaml

      name: apache2
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: update repository index
    dnf:
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: install apache2 package
    dnf:
      name: httpd
      state: latest
    when: ansible_distribution == "CentOS"

  - name: add PHP support for apache
    dnf:
```



```
      name: php
      state: latest
    when: ansible_distribution == "CentOS"
```

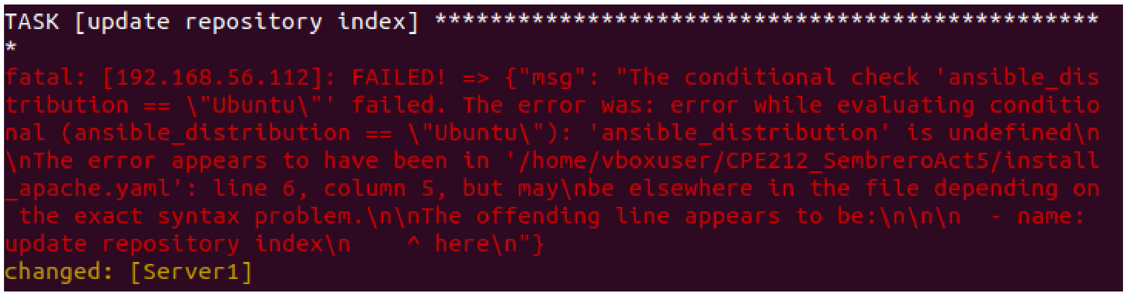
Make sure to save and exit.

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
TASK [install apache2 package] ************************************************
*
ok: [Server1]

TASK [add PHP support for apache] *********************************************
*
ok: [Server1]

TASK [update repository index] ************************************************
*
skipping: [Server1]

TASK [install apache2 package] ************************************************
*
skipping: [Server1]

TASK [add PHP support for apache] *********************************************
*
skipping: [Server1]
        to retry, use: --limit @/home/vboxuser/CPE212_SembreroAct5/install_apac
he.retry

PLAY RECAP ********************************************************************
*
192.168.56.112             : ok=1    changed=0    unreachable=0    failed=1
Server1                    : ok=4    changed=1    unreachable=0    failed=0
```

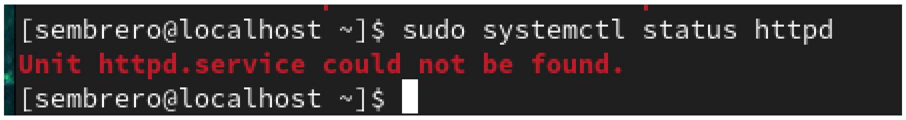
CentOS still failed:



```
TASK [update repository index] ***************************************************
*
fatal: [192.168.56.112]: FAILED! => {"msg": "The conditional check 'ansible_dis
tribution == \"Ubuntu\"' failed. The error was: error while evaluating conditio
nal (ansible_distribution == \"Ubuntu\"): 'ansible_distribution' is undefined\n
\nThe error appears to have been in '/home/vboxuser/CPE212_SembreroAct5/install
_apache.yaml': line 6, column 5, but may\nbe elsewhere in the file depending on
 the exact syntax problem.\n\nThe offending line appears to be:\n\n\n  - name:
update repository index\n    ^ here\n"}
changed: [Server1]
```

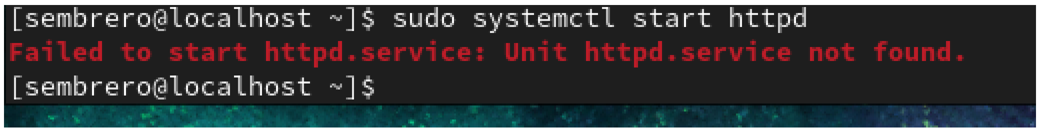
5. To verify the installations, go to CentOS VM and type its IP address on the browser. Was it successful? The answer is no. It's because the httpd service or the Apache HTTP server in the CentOS is not yet active. Thus, you need to activate it first.

   5.1 To activate, go to the CentOS VM terminal and enter the following:
   systemctl status httpd
   The result of this command tells you that the service is inactive.



```
[sembrero@localhost ~]$ sudo systemctl status httpd
Unit httpd.service could not be found.
[sembrero@localhost ~]$
```

   5.2 Issue the following command to start the service:
   sudo systemctl start httpd
   (When prompted, enter the sudo password)
   sudo firewall-cmd --add-port=80/tcp
   (The result should be a success)

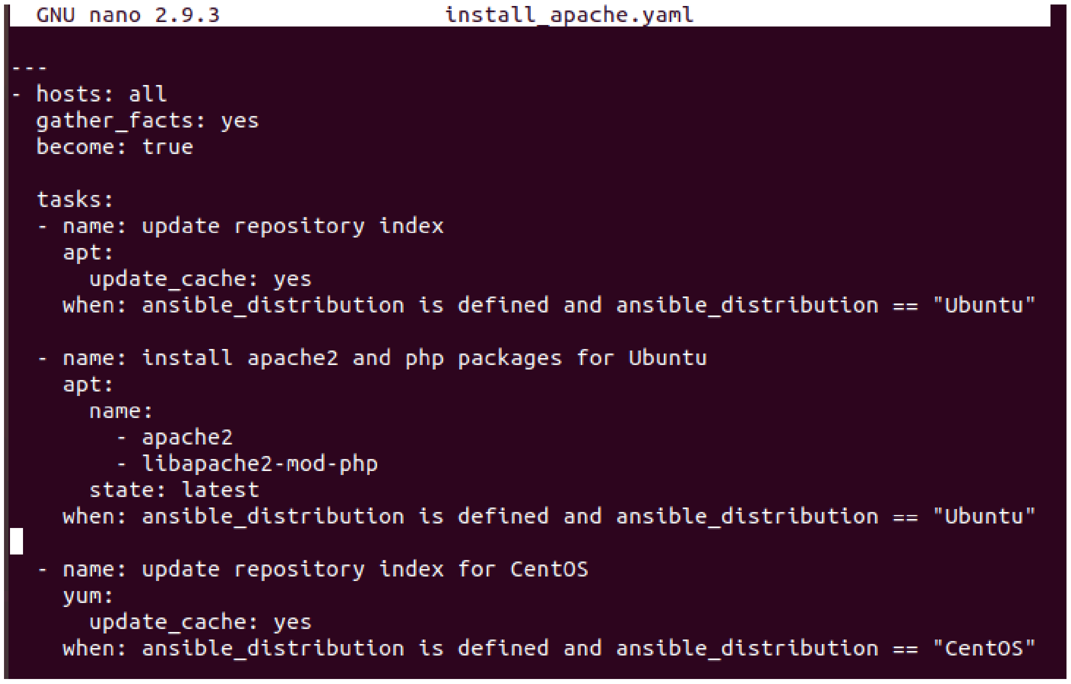


```
[sembrero@localhost ~]$ sudo systemctl start httpd
Failed to start httpd.service: Unit httpd.service not found.
[sembrero@localhost ~]$
```

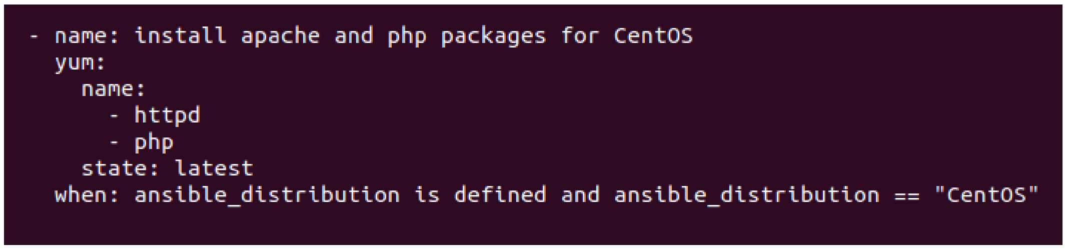
   5.3 To verify the service is already running, go to CentOS VM and type its IP address on the browser. Was it successful? (Screenshot the browser)

Task 2: Refactoring playbook
This time, we want to make sure that our playbook is efficient and that the codes are easier to read. This will also makes run ansible more quickly if it has to execute fewer tasks to do the same thing.

1. Edit the playbook *install_apache.yml*. Currently, we have three tasks targeting our Ubuntu machines and 3 tasks targeting our CentOS machine. Right now, we try to consolidate some tasks that are typically the same. For example, we can consolidate two plays that install packages. We can do that by creating a list of installation packages as shown below:



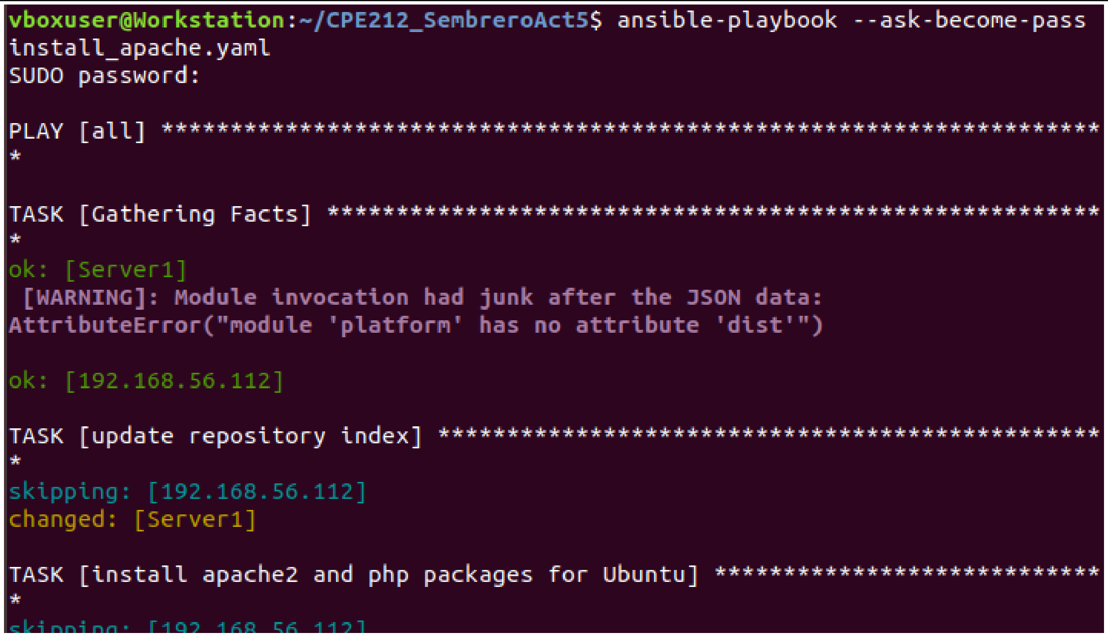
```
  GNU nano 2.9.3                  install_apache.yaml

---
- hosts: all
  gather_facts: yes
  become: true

  tasks:
  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution is defined and ansible_distribution == "Ubuntu"

  - name: install apache2 and php packages for Ubuntu
    apt:
      name:
        - apache2
        - libapache2-mod-php
      state: latest
    when: ansible_distribution is defined and ansible_distribution == "Ubuntu"

  - name: update repository index for CentOS
    yum:
      update_cache: yes
    when: ansible_distribution is defined and ansible_distribution == "CentOS"
```



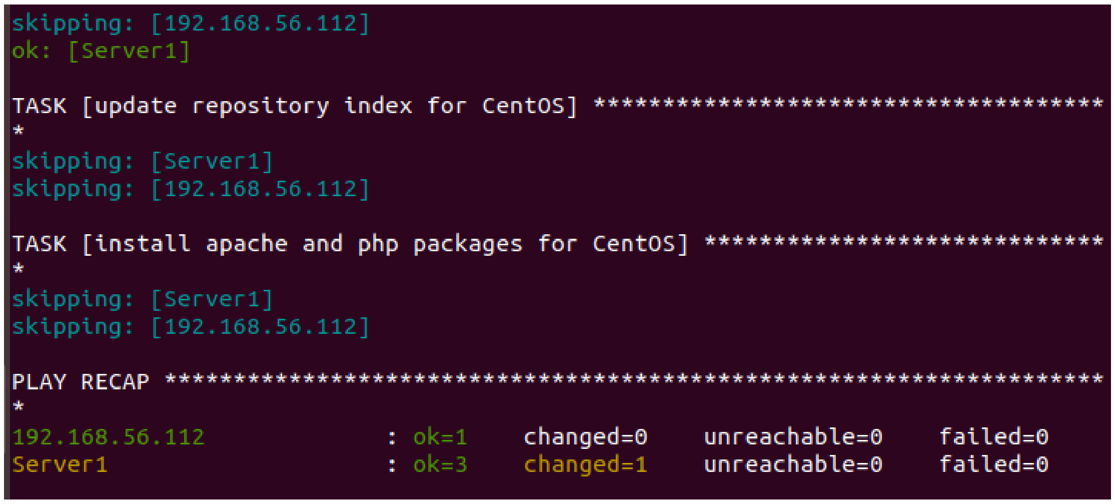
```
  - name: install apache and php packages for CentOS
    yum:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution is defined and ansible_distribution == "CentOS"
```

Make sure to save the file and exit.

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



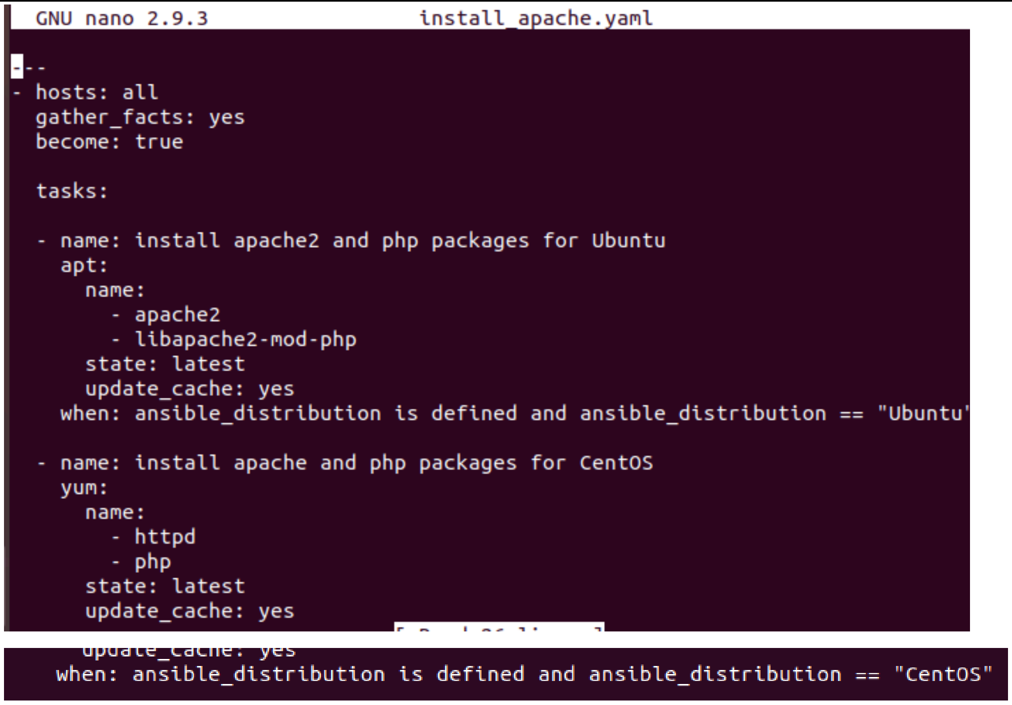
```
vboxuser@Workstation:~/CPE212_SembreroAct5$ ansible-playbook --ask-become-pass
install_apache.yaml
SUDO password:

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

TASK [Gathering Facts] *********************************************************
*
ok: [Server1]
 [WARNING]: Module invocation had junk after the JSON data:
AttributeError("module 'platform' has no attribute 'dist'")

ok: [192.168.56.112]

TASK [update repository index] *************************************************
*
skipping: [192.168.56.112]
changed: [Server1]

TASK [install apache2 and php packages for Ubuntu] *****************************
*
skipping: [192.168.56.112]
```



```
skipping: [192.168.56.112]
ok: [Server1]

TASK [update repository index for CentOS] **************************************
*
skipping: [Server1]
skipping: [192.168.56.112]

TASK [install apache and php packages for CentOS] ******************************
*
skipping: [Server1]
skipping: [192.168.56.112]

PLAY RECAP *********************************************************************
*
192.168.56.112             : ok=1    changed=0    unreachable=0    failed=0
Server1                    : ok=3    changed=1    unreachable=0    failed=0
```

Finally, after a bunch of trial and error, I have successfully include CentOS for installation using playbook.

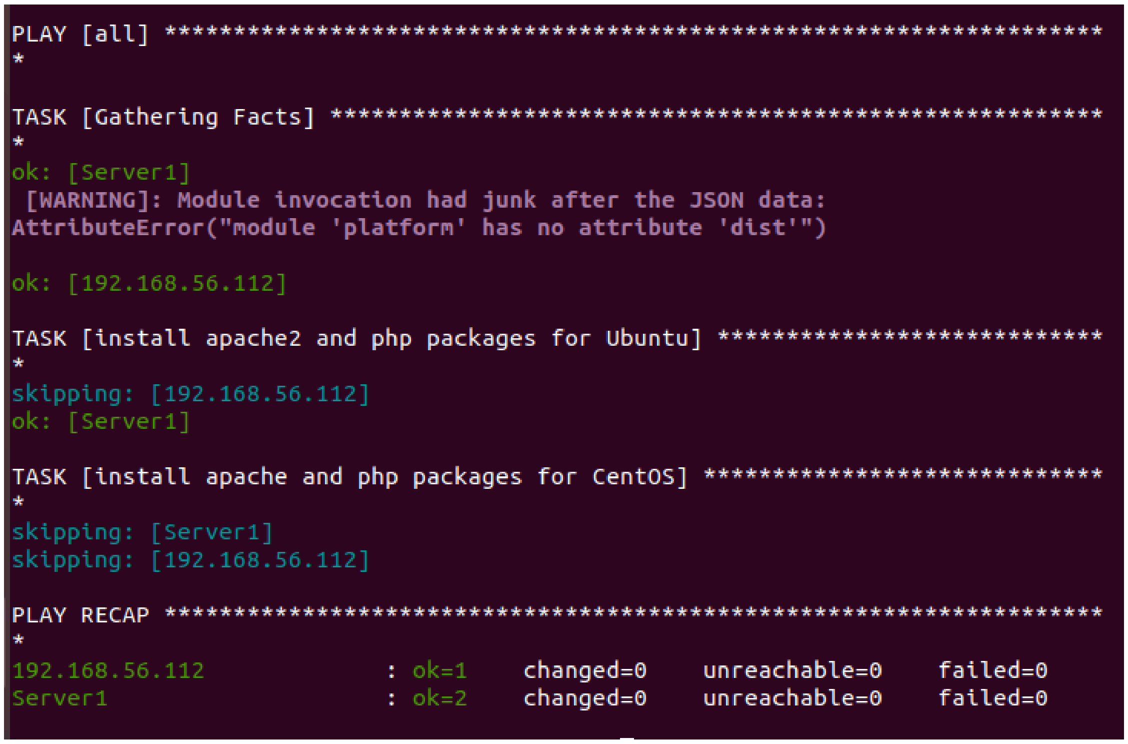
2. Edit the playbook *install_apache.yml* again. In task 2.1, we consolidated the plays into one play. This time we can actually consolidated everything in just 2 plays. This can be done by removing the update repository play and putting the command *update_cache: yes* below the command *state: latest*. See below for reference:



```
  GNU nano 2.9.3                    install_apache.yaml

---
- hosts: all
  gather_facts: yes
  become: true

  tasks:

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

  - name: install apache and php packages for CentOS
    yum:
      name:
        - httpd
        - php
      state: latest
      update_cache: yes
    when: ansible_distribution is defined and ansible_distribution == "CentOS"
```

Make sure to save the file and exit.

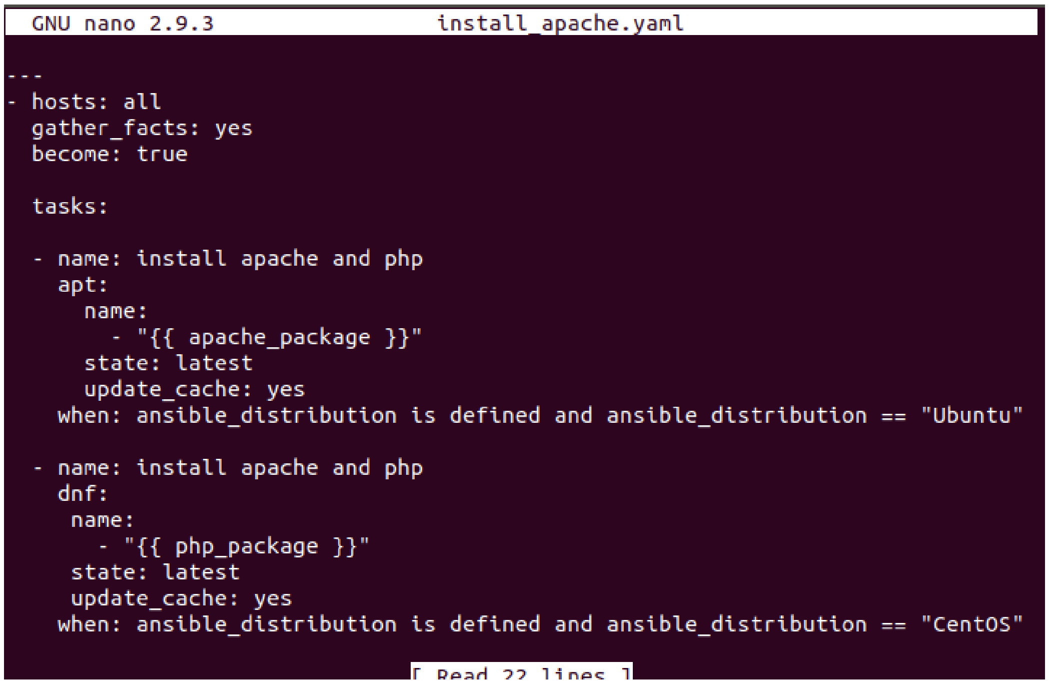
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



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

TASK [Gathering Facts] *******************************************************
*
ok: [Server1]
 [WARNING]: Module invocation had junk after the JSON data:
AttributeError("module 'platform' has no attribute 'dist'")

ok: [192.168.56.112]

TASK [install apache2 and php packages for Ubuntu] ***************************
*
skipping: [192.168.56.112]
ok: [Server1]

TASK [install apache and php packages for CentOS] ****************************
*
skipping: [Server1]
skipping: [192.168.56.112]

PLAY RECAP *******************************************************************
*
192.168.56.112             : ok=1    changed=0    unreachable=0    failed=0
Server1                    : ok=2    changed=0    unreachable=0    failed=0
```

Results become shorter because updating the repository index was removed from the playbook. The update_cache:yes from that task was placed for remaining tasks of ubuntu and centos.

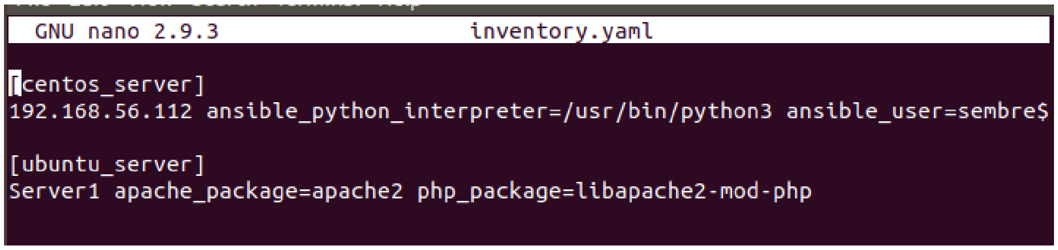
3. Finally, we can consolidate these 2 plays in just 1 play. This can be done by declaring variables that will represent the packages that we want to install. Basically, the apache_package and php_package are variables. The names are arbitrary, which means we can choose different names. We also take out the line when: ansible_distribution. Edit the playbook *install_apache.yml* again and make sure to follow the below image. Make sure to save the file and exit.



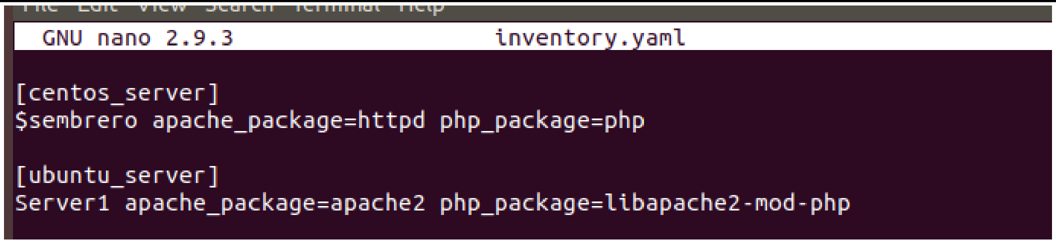
```
  GNU nano 2.9.3                    install_apache.yaml

---
- hosts: all
  gather_facts: yes
  become: true

  tasks:

  - name: install apache and php
    apt:
      name:
        - "{{ apache_package }}"
      state: latest
      update_cache: yes
    when: ansible_distribution is defined and ansible_distribution == "Ubuntu"

  - name: install apache and php
    dnf:
     name:
       - "{{ php_package }}"
     state: latest
     update_cache: yes
    when: ansible_distribution is defined and ansible_distribution == "CentOS"

                               [ Read 22 lines ]
```

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.

4. Unfortunately, task 2.3 was not successful. It's because we need to change something in the inventory file so that the variables we declared will be in place. Edit the *inventory* file and follow the below configuration:



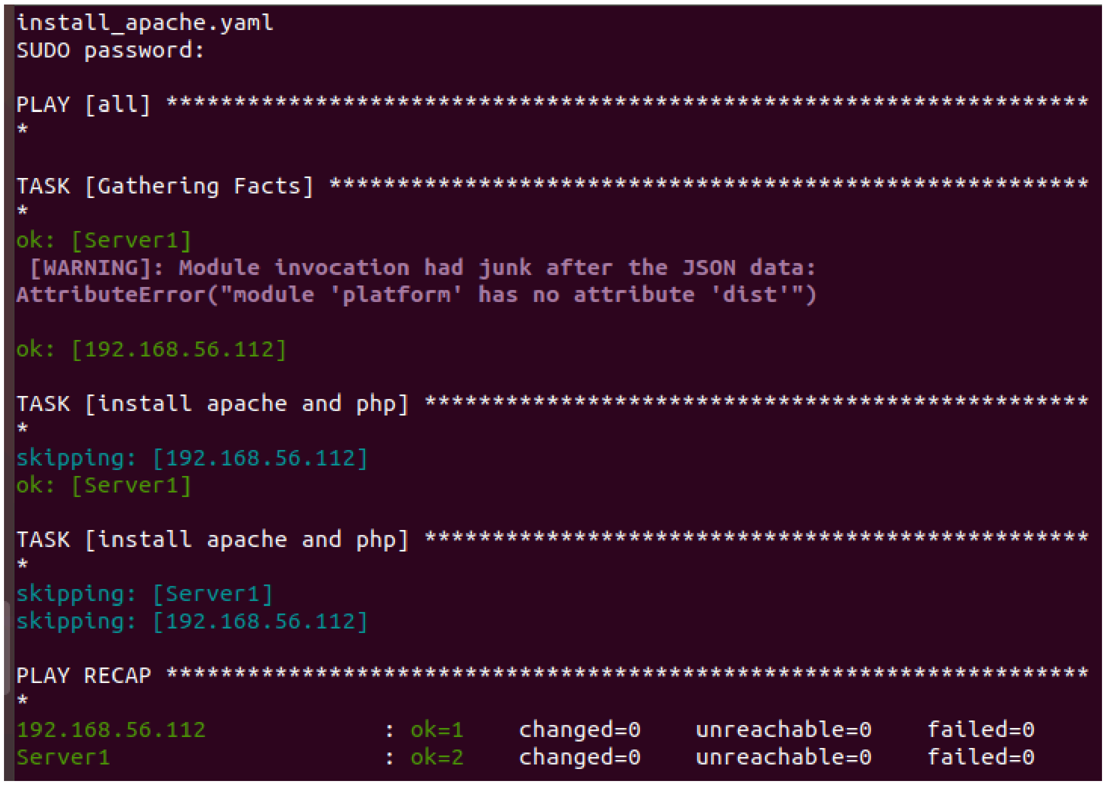
```
  GNU nano 2.9.3                    inventory.yaml

[centos_server]
192.168.56.112 ansible_python_interpreter=/usr/bin/python3 ansible_user=sembre$

[ubuntu_server]
Server1 apache_package=apache2 php_package=libapache2-mod-php
```


```
 GNU nano 2.9.3                          inventory.yaml

[centos_server]
$sembrero apache_package=httpd php_package=php

[ubuntu_server]
Server1 apache_package=apache2 php_package=libapache2-mod-php
```

Make sure to save the *inventory* file and exit.

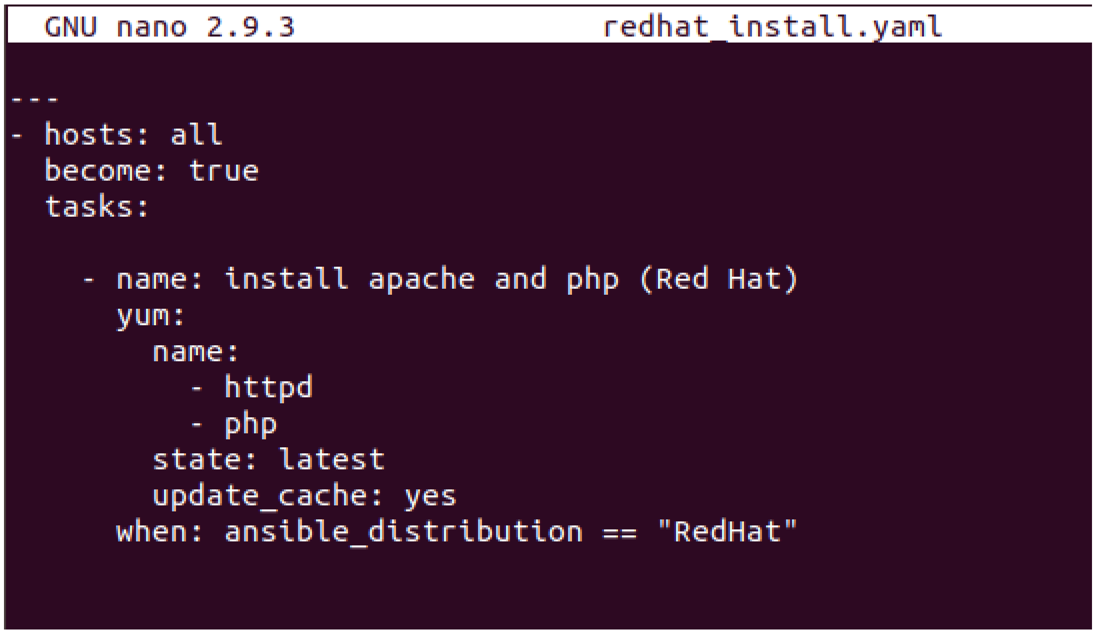
Finally, we still have one more thing to change in our *install_apache.yml* file. In task 2.3, you may notice that the package is assign as apt, which will not run in CentOS. Replace the *apt* with *package*. Package is a module in ansible that is generic, which is going to use whatever package manager the underlying host or the target server uses. For Ubuntu it will automatically use *apt*, and for CentOS it will automatically use *dnf*. Make sure to save the file and exit. For more details about the ansible package, you may refer to this documentation: [ansible.builtin.package – Generic OS package manager — Ansible Documentation]()

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.

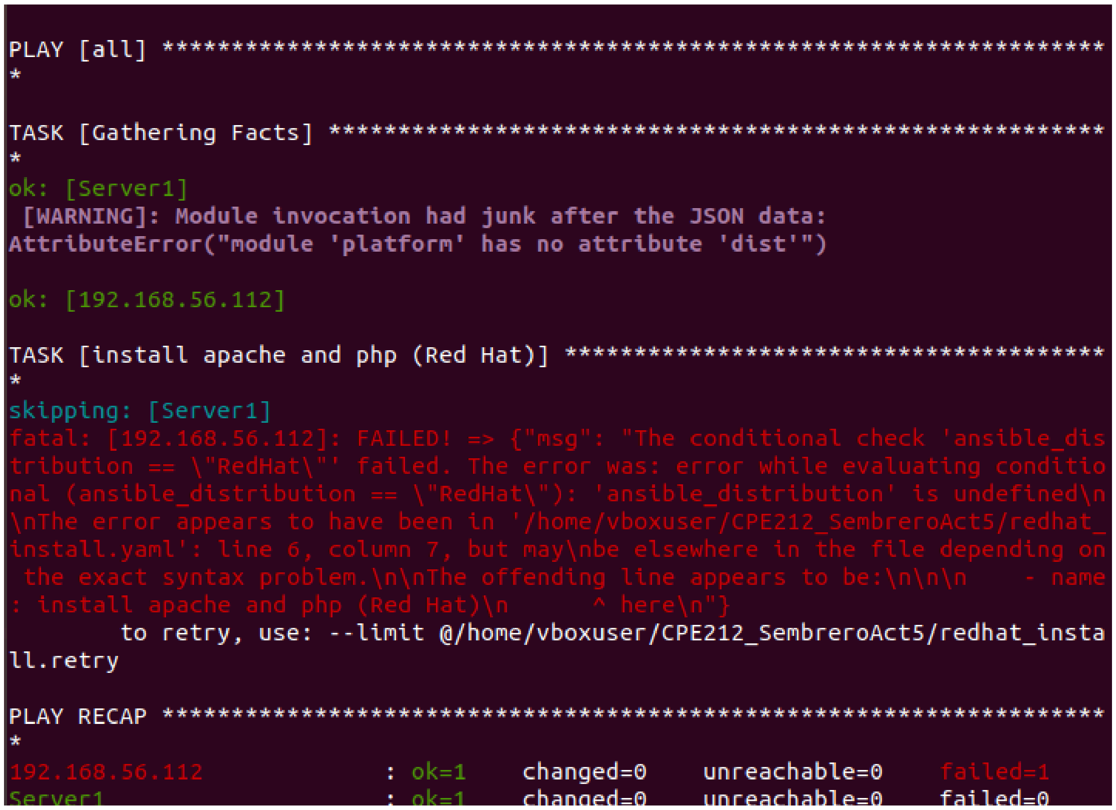

```
install_apache.yaml
SUDO password:

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

TASK [Gathering Facts] ********************************************************
*
ok: [Server1]
 [WARNING]: Module invocation had junk after the JSON data:
AttributeError("module 'platform' has no attribute 'dist'")

ok: [192.168.56.112]

TASK [install apache and php] *************************************************
*
skipping: [192.168.56.112]
ok: [Server1]

TASK [install apache and php] *************************************************
*
skipping: [Server1]
skipping: [192.168.56.112]

PLAY RECAP ********************************************************************
*
192.168.56.112             : ok=1    changed=0    unreachable=0    failed=0
Server1                    : ok=2    changed=0    unreachable=0    failed=0
```

Supplementary Activity:

1. Create a playbook that could do the previous tasks in Red Hat OS.



```
  GNU nano 2.9.3                         redhat_install.yaml

---
- hosts: all
  become: true
  tasks:

    - name: install apache and php (Red Hat)
      yum:
        name:
          - httpd
          - php
        state: latest
        update_cache: yes
      when: ansible_distribution == "RedHat"
```



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

TASK [Gathering Facts] ********************************************************
*
ok: [Server1]
 [WARNING]: Module invocation had junk after the JSON data:
AttributeError("module 'platform' has no attribute 'dist'")

ok: [192.168.56.112]

TASK [install apache and php (Red Hat)] ***************************************
*
skipping: [Server1]
fatal: [192.168.56.112]: FAILED! => {"msg": "The conditional check 'ansible_dis
tribution == \"RedHat\"' failed. The error was: error while evaluating conditio
nal (ansible_distribution == \"RedHat\"): 'ansible_distribution' is undefined\n
\nThe error appears to have been in '/home/vboxuser/CPE212_SembreroAct5/redhat_
install.yaml': line 6, column 7, but may\nbe elsewhere in the file depending on
 the exact syntax problem.\n\nThe offending line appears to be:\n\n    - name
: install apache and php (Red Hat)\n      ^ here\n"}
        to retry, use: --limit @/home/vboxuser/CPE212_SembreroAct5/redhat_insta
ll.retry

PLAY RECAP ********************************************************************
*
192.168.56.112             : ok=1    changed=0    unreachable=0    failed=1
Server1                    : ok=1    changed=0    unreachable=0    failed=0
```

Reflections:
Answer the following:

1. **Why do you think refactoring of playbook codes is important?**
   When you refactor, you make the playbook easier to read, maintain, and maintain. You eliminate repeating the same code, which allows the team to collaborate more efficiently. You are less prone to make mistakes, and everything is easier to use again later.

2. **When do we use the "when" command in playbook?**

   Use when to only run a task if a certain condition is true, like inspecting the OS or checking a variable. In this way, playbooks will do what is necessary on each machine.