Name: Gabiano, Chris Leonard A.	**Date Performed: Sept 18, 2023**
Course/Section: CPE231s6	**Date Submitted: Sept 15, 2023**
Instructor: Dr. Jonathan Taylar	**Semester and SY: 2023-24**

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.

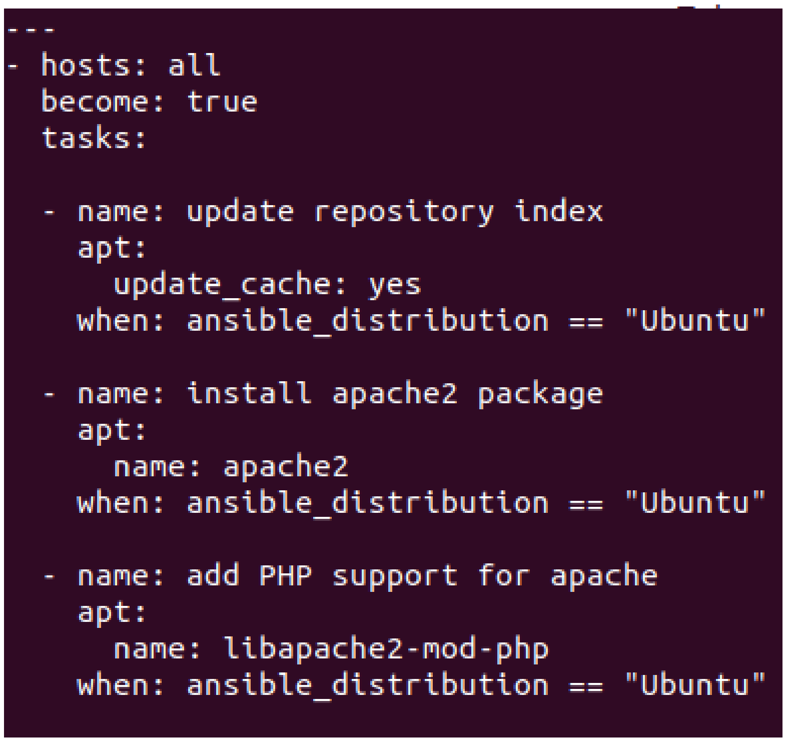
Requirement:
In this activity, you will need to create a CentOS VM. Likewise, you need to activate the second adapter to a host-only adapter after the installations. Take note of the IP address of the CentOS VM. Make sure to use the command *ssh-copy-id* to copy the public key to CentOS. Verify if you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

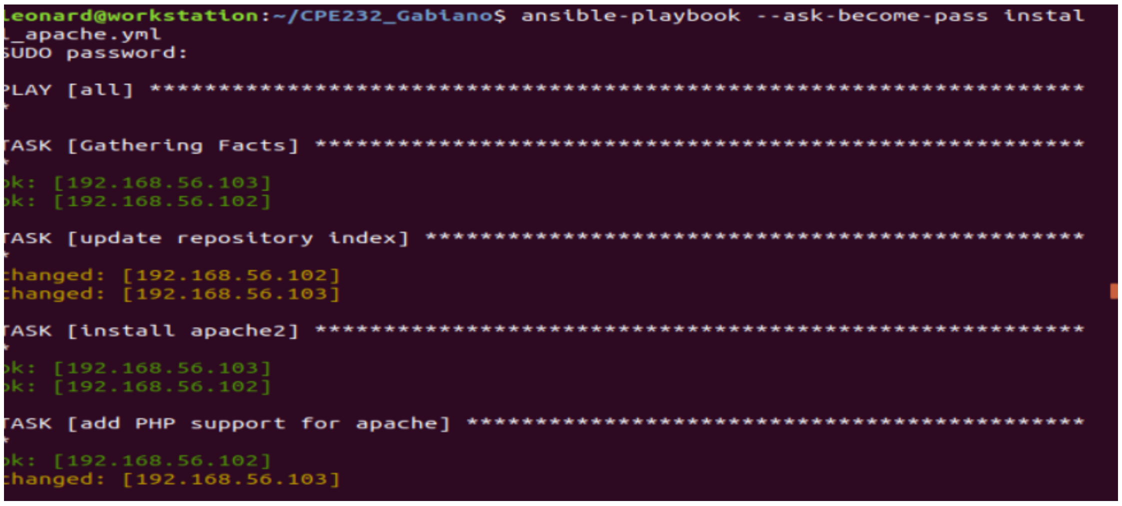
3. Edit the *install_apache.yml* file and insert the lines shown below.



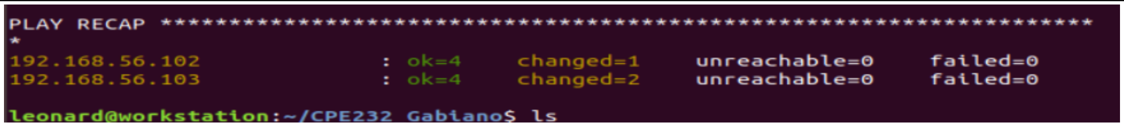
```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution == "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.



```
leonard@workstation:~/CPE232_Gabiano$ ansible-playbook --ask-become-pass instal
_apache.yml
UDO password:

LAY [all] **********************************************************************
*

ASK [Gathering Facts] **********************************************************
*
k: [192.168.56.103]
k: [192.168.56.102]

ASK [update repository index] **************************************************
*
hanged: [192.168.56.102]
hanged: [192.168.56.103]

ASK [install apache2] **********************************************************
*
k: [192.168.56.103]
k: [192.168.56.102]

ASK [add PHP support for apache] ***********************************************
*
k: [192.168.56.102]
hanged: [192.168.56.103]
```



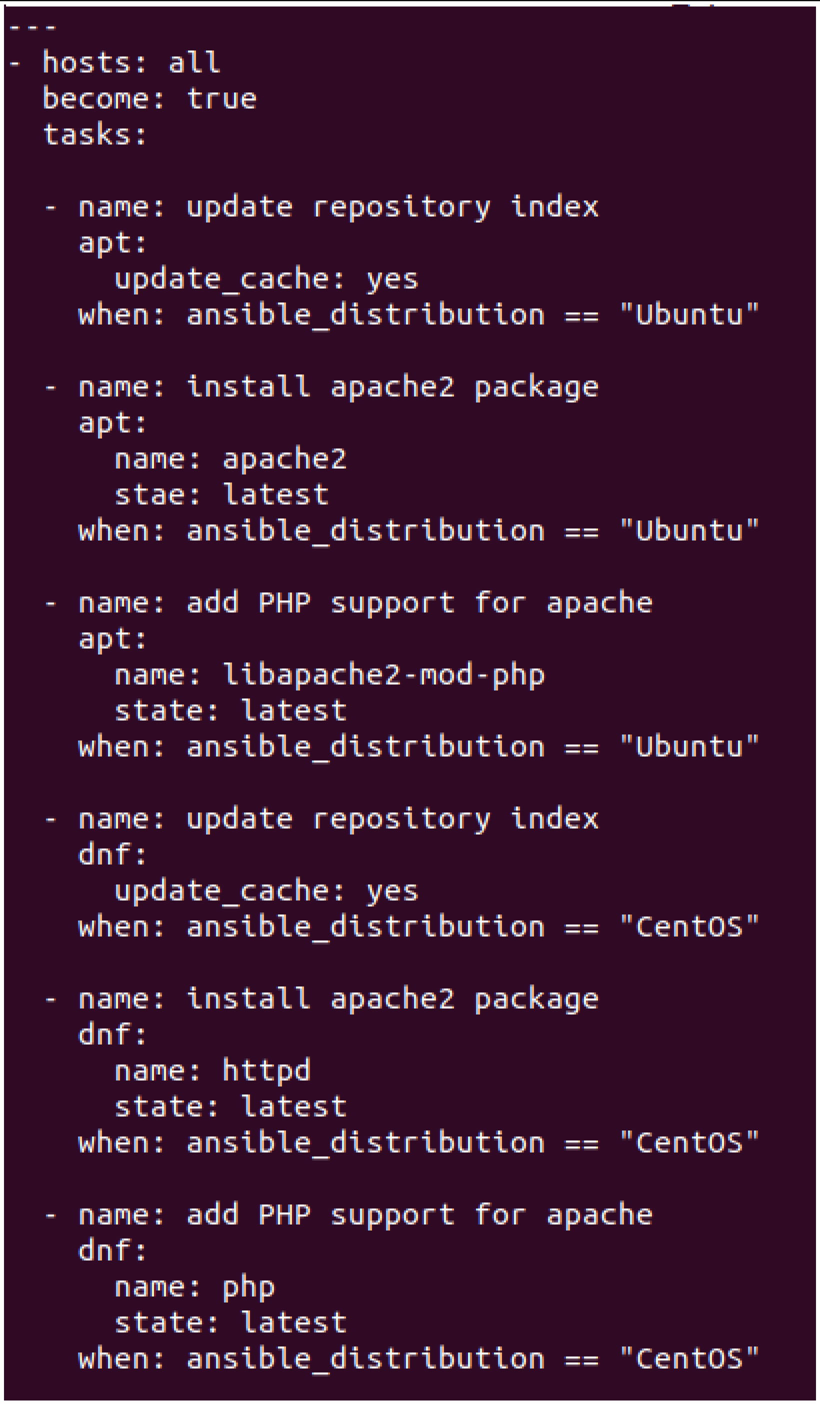
```
PLAY RECAP **********************************************************************
*
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0
192.168.56.103             : ok=4    changed=2    unreachable=0    failed=0

leonard@workstation:~/CPE232_Gabiano$ ls
```

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]
```

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

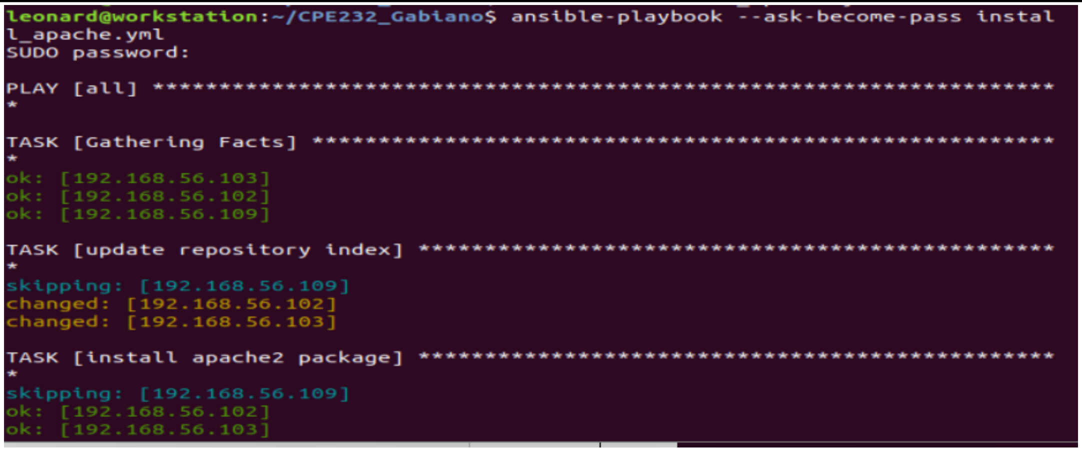
4. Edit the *install_apache.yml* file and insert the lines shown below.



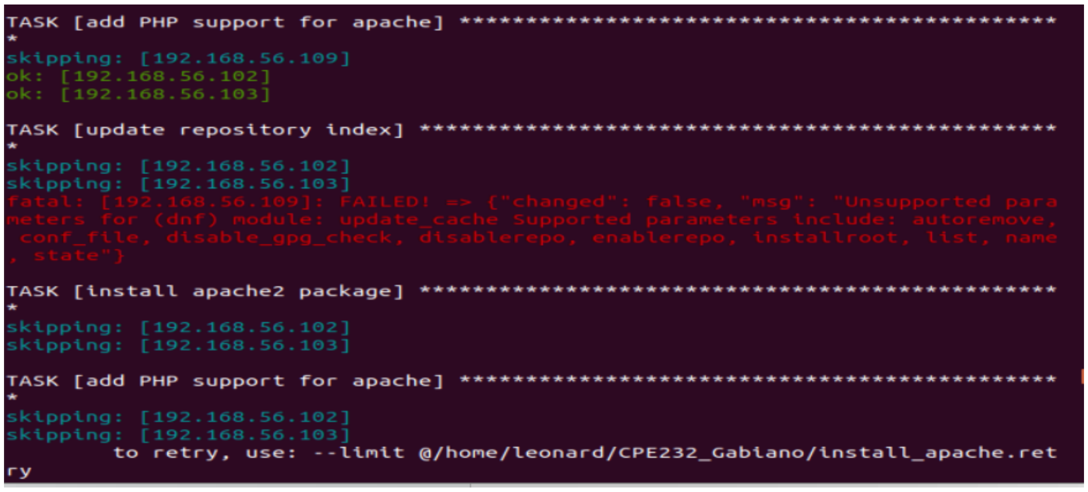
```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

  - name: install apache2 package
    apt:
      name: apache2
      stae: 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.

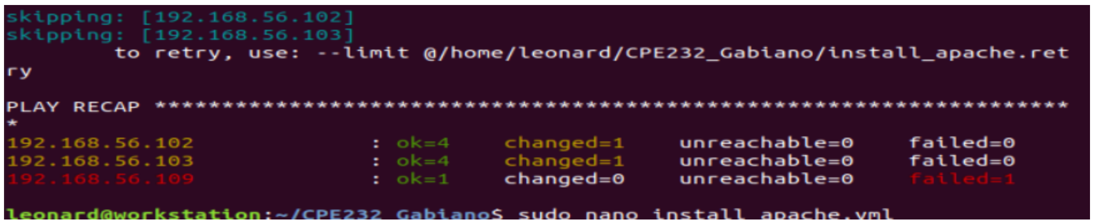
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
leonard@workstation:~/CPE232_Gabiano$ ansible-playbook --ask-become-pass instal
l_apache.yml
SUDO password:

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

TASK [Gathering Facts] **********************************************************
*
ok: [192.168.56.103]
ok: [192.168.56.102]
ok: [192.168.56.109]

TASK [update repository index] **************************************************
*
skipping: [192.168.56.109]
changed: [192.168.56.102]
changed: [192.168.56.103]

TASK [install apache2 package] **************************************************
*
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]
```

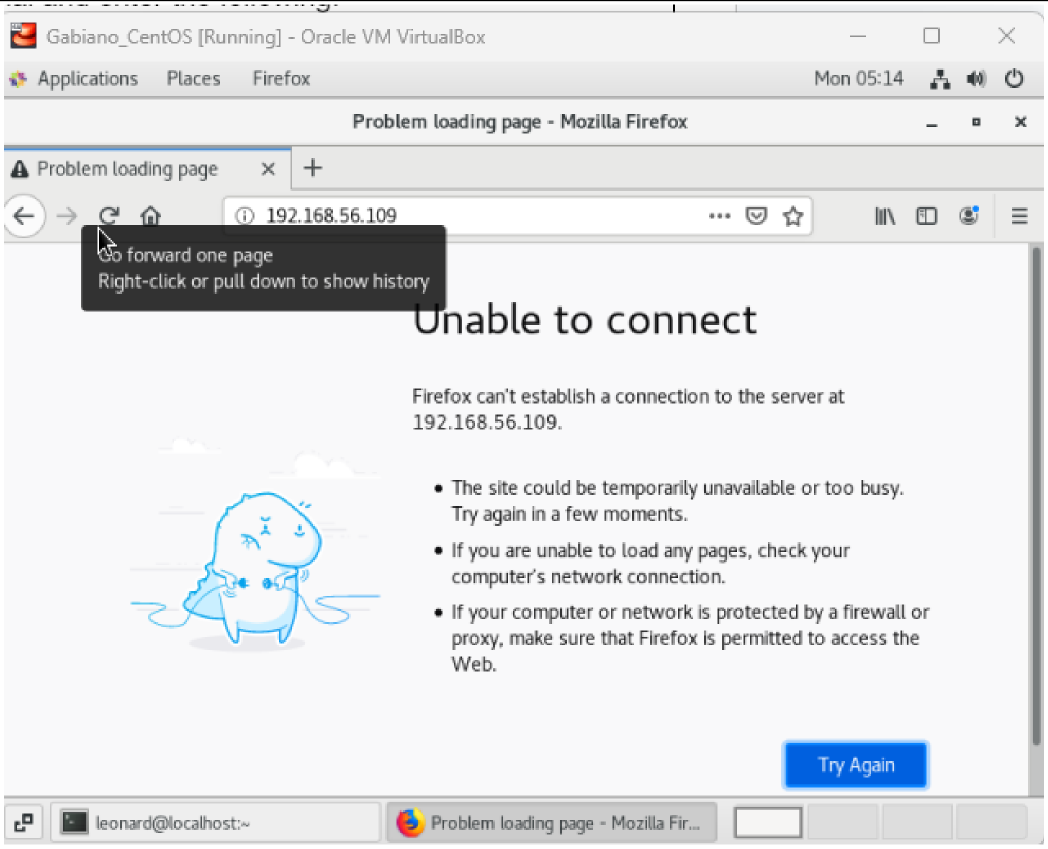


```
TASK [add PHP support for apache] ***********************************************
*
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [update repository index] **************************************************
*
skipping: [192.168.56.102]
skipping: [192.168.56.103]
fatal: [192.168.56.109]: FAILED! => {"changed": false, "msg": "Unsupported para
meters for (dnf) module: update_cache Supported parameters include: autoremove,
 conf_file, disable_gpg_check, disablerepo, enablerepo, installroot, list, name
, state"}

TASK [install apache2 package] **************************************************
*
skipping: [192.168.56.102]
skipping: [192.168.56.103]

TASK [add PHP support for apache] ***********************************************
*
skipping: [192.168.56.102]
skipping: [192.168.56.103]
        to retry, use: --limit @/home/leonard/CPE232_Gabiano/install_apache.ret
ry
```



```
skipping: [192.168.56.102]
skipping: [192.168.56.103]
        to retry, use: --limit @/home/leonard/CPE232_Gabiano/install_apache.ret
ry

PLAY RECAP **********************************************************************
*
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0
192.168.56.103             : ok=4    changed=1    unreachable=0    failed=0
192.168.56.109             : ok=1    changed=0    unreachable=0    failed=1

leonard@workstation:~/CPE232_Gabiano$ sudo nano install_apache.yml
```

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.


Gabiano_CentOS [Running] - Oracle VM VirtualBox
Applications
Places
Firefox
Mon 05:14
Problem loading page - Mozilla Firefox
Problem loading page
192.168.56.109
Go forward one page
Right-click or pull down to show history
Try Again
leonard@localhost:~
Problem loading page - Mozilla Fir...

Unable to connect

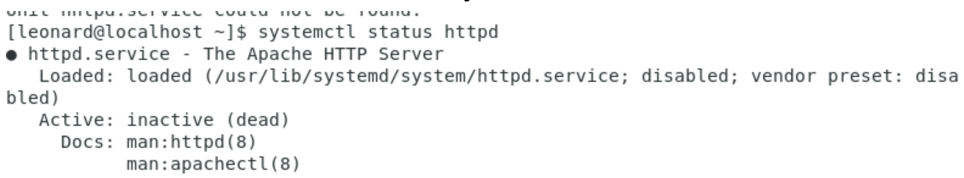
Firefox can't establish a connection to the server at 192.168.56.109.

- 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.

5.1 To activate, go to the CentOS VM terminal and enter the following:

systemctl status httpd

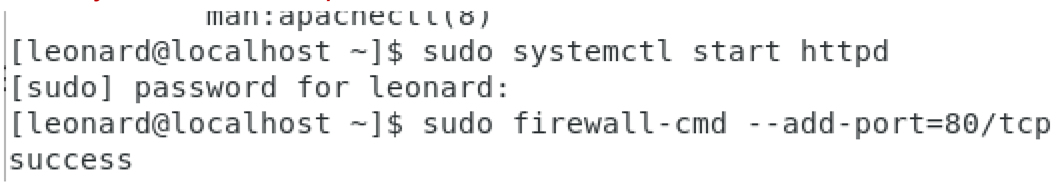
The result of this command tells you that the service is inactive.



```
[leonard@localhost ~]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disa
bled)
   Active: inactive (dead)
     Docs: man:httpd(8)
           man:apachectl(8)
```

5.2 Issue the following command to start the service:

sudo systemctl start httpd

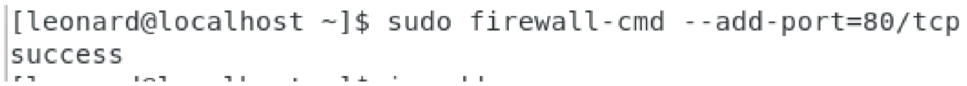


```
[leonard@localhost ~]$ sudo systemctl start httpd
[sudo] password for leonard:
[leonard@localhost ~]$ sudo firewall-cmd --add-port=80/tcp
success
```

(When prompted, enter the sudo password)

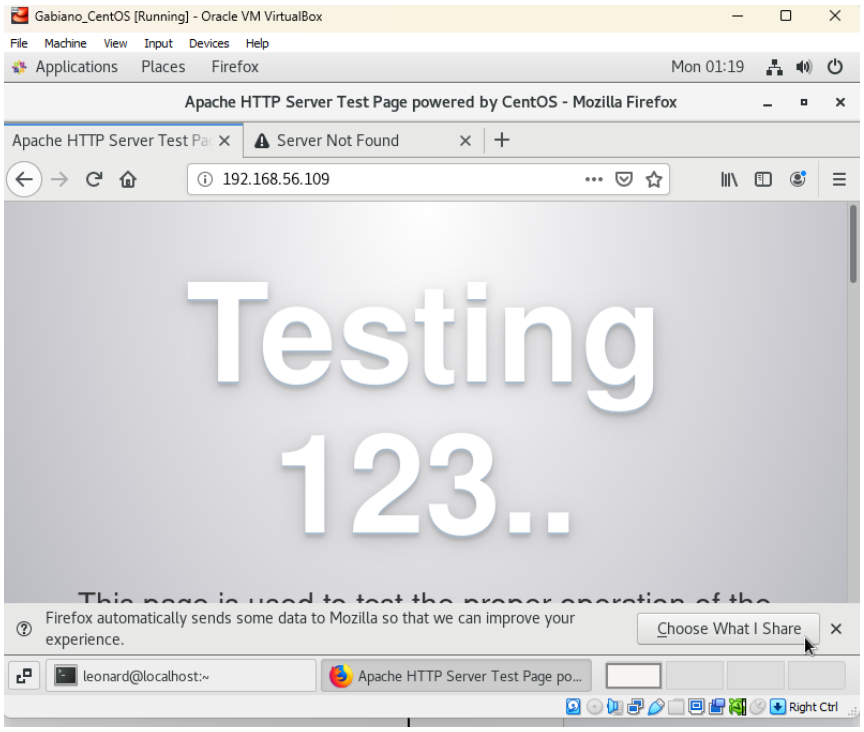
sudo firewall-cmd --add-port=80/tcp

(The result should be a success)



```
[leonard@localhost ~]$ sudo firewall-cmd --add-port=80/tcp
success
```

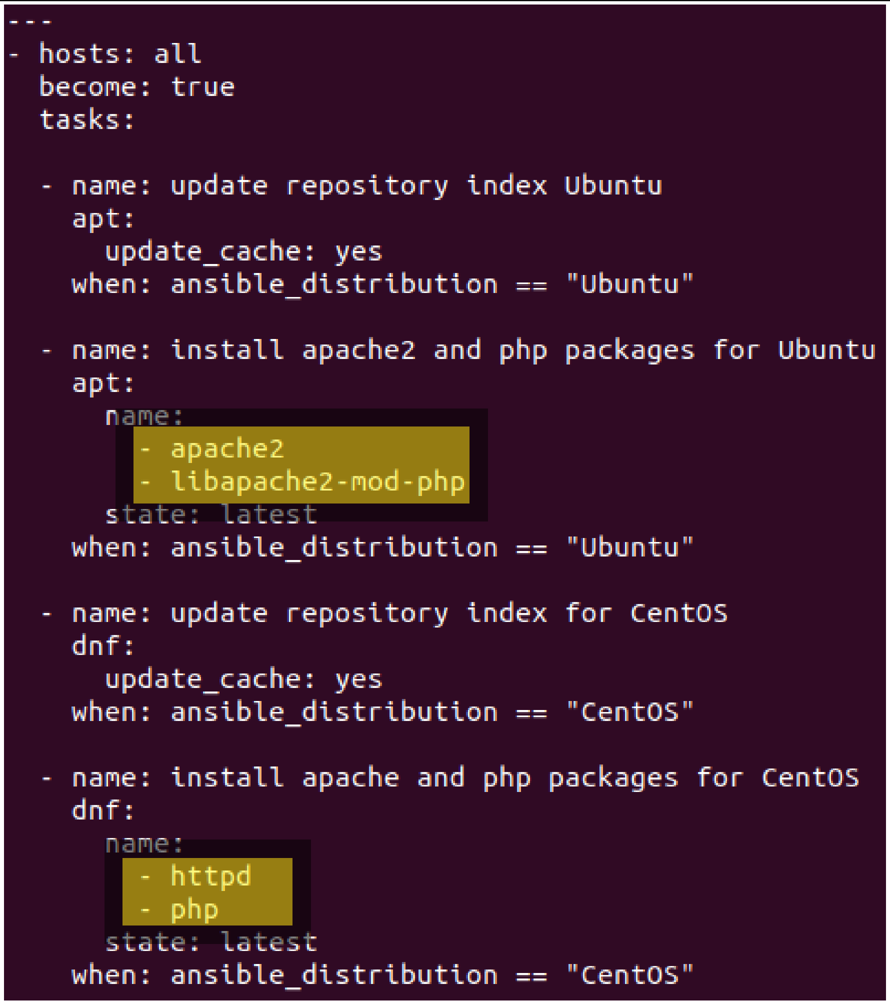
5.3 To verify the service is already running, go to CentOS VM and type its IP address on the browser. Was it successful? (Screenshot the browser)


Gabiano_CentOS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Firefox
Mon 01:19
Apache HTTP Server Test Page powered by CentOS - Mozilla Firefox
Apache HTTP Server Test Pa
Server Not Found
192.168.56.109
Testing 123..
This page is used to test the proper operation of the
Firefox automatically sends some data to Mozilla so that we can improve your experience.
Choose What I Share
leonard@localhost:~
Apache HTTP Server Test Page po...
Right Ctrl

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.

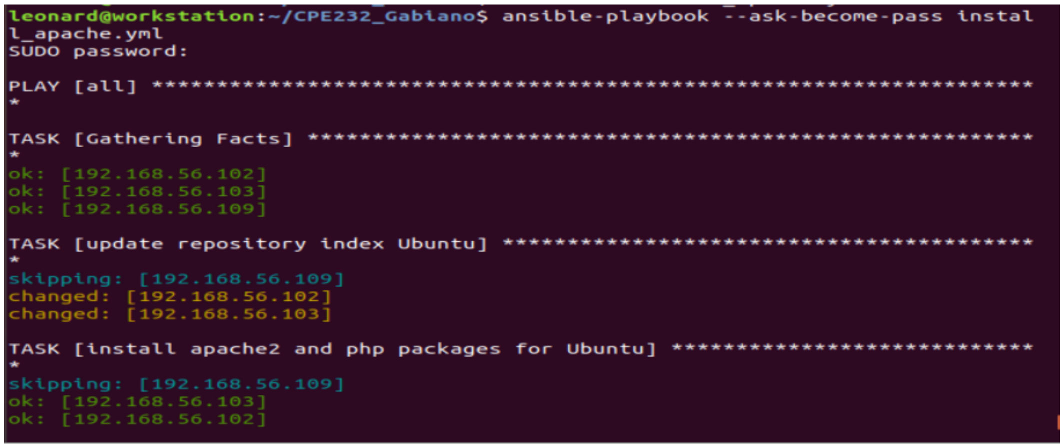
1. Edit the playbook *install_apache.yml*. Currently, we have three tasks targeting our Ubuntu machines and 3 tasks targeting our CentOS machine. Right now, we try to consolidate some tasks that are typically the same. For example, we can consolidate two plays that install packages. We can do that by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



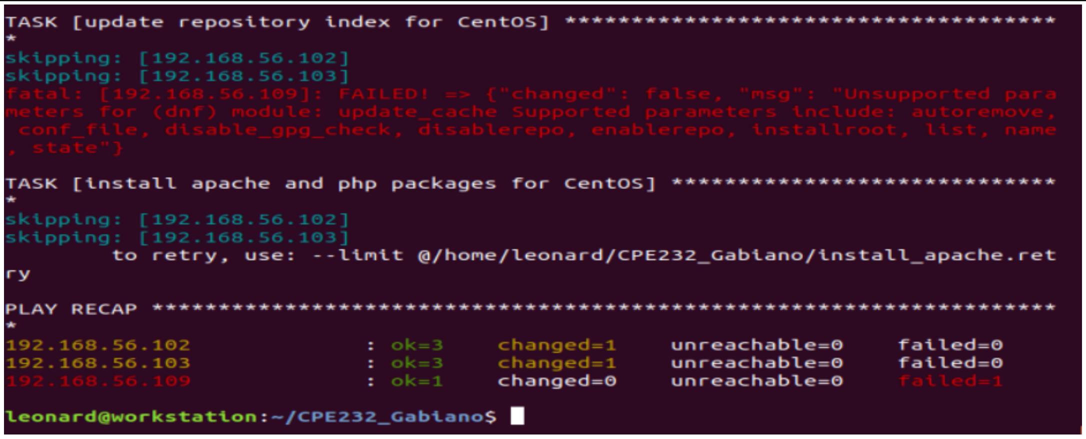
```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index Ubuntu
    apt:
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

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

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

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

Make sure to save the file and exit.

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

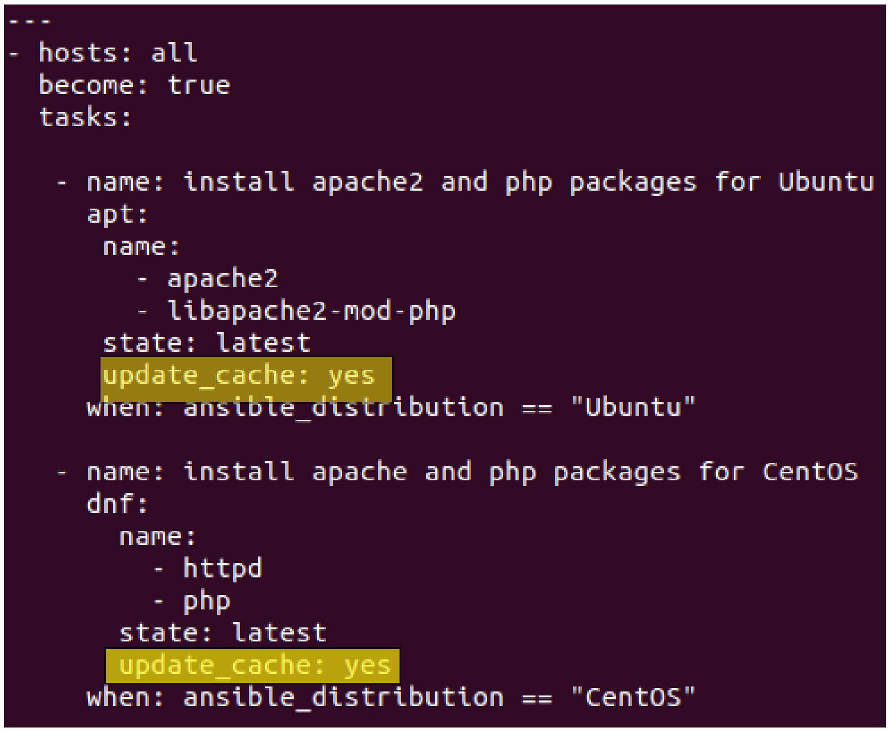


```
leonard@workstation:~/CPE232_Gabiano$ ansible-playbook --ask-become-pass instal
l_apache.yml
SUDO password:

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

TASK [Gathering Facts] **********************************************************
*
ok: [192.168.56.102]
ok: [192.168.56.103]
ok: [192.168.56.109]

TASK [update repository index Ubuntu] *******************************************
*
skipping: [192.168.56.109]
changed: [192.168.56.102]
changed: [192.168.56.103]

TASK [install apache2 and php packages for Ubuntu] ******************************
*
skipping: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]
```



```
TASK [update repository index for CentOS] ***************************************
*
skipping: [192.168.56.102]
skipping: [192.168.56.103]
fatal: [192.168.56.109]: FAILED! => {"changed": false, "msg": "Unsupported para
meters for (dnf) module: update_cache Supported parameters include: autoremove,
 conf_file, disable_gpg_check, disablerepo, enablerepo, installroot, list, name
, state"}

TASK [install apache and php packages for CentOS] *******************************
*
skipping: [192.168.56.102]
skipping: [192.168.56.103]
        to retry, use: --limit @/home/leonard/CPE232_Gabiano/install_apache.ret
ry

PLAY RECAP **********************************************************************
*
192.168.56.102             : ok=3    changed=1    unreachable=0    failed=0
192.168.56.103             : ok=3    changed=1    unreachable=0    failed=0
192.168.56.109             : ok=1    changed=0    unreachable=0    failed=1

leonard@workstation:~/CPE232_Gabiano$ 
```

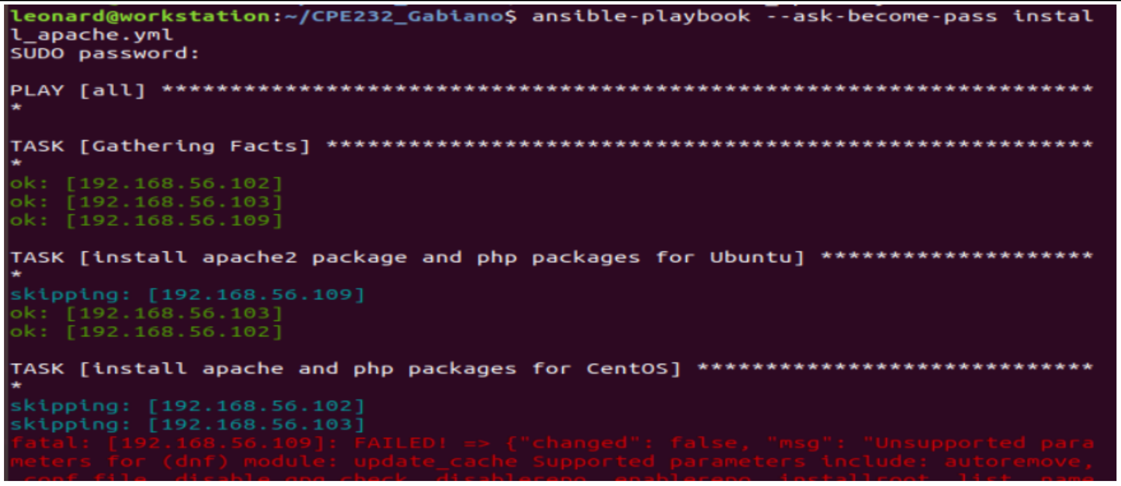
2. Edit the playbook *install_apache.yml* again. In task 2.1, we consolidated the plays into one play. This time we can actually consolidated everything in just 2 plays. This can be done by removing the update repository play and putting the command *update_cache: yes* below the command *state: latest*.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



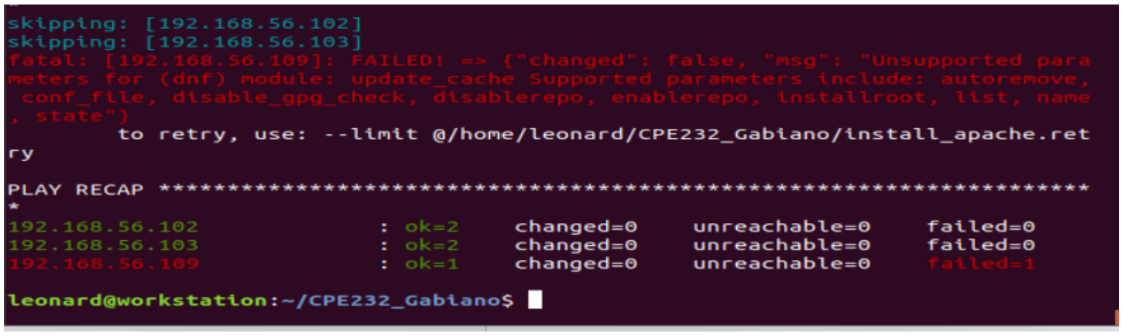
```
---
- hosts: all
  become: true
  tasks:

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

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

Make sure to save the file and exit.

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

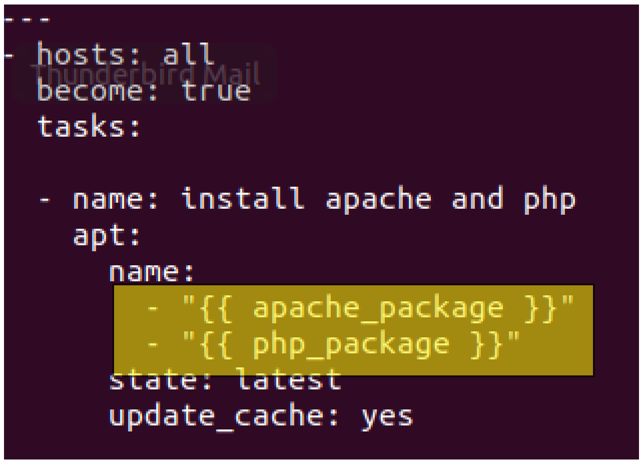


```
leonard@workstation:~/CPE232_Gabiano$ ansible-playbook --ask-become-pass instal
l_apache.yml
SUDO password:

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

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.102]
ok: [192.168.56.103]
ok: [192.168.56.109]

TASK [install apache2 package and php packages for Ubuntu] ********************
*
skipping: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]

TASK [install apache and php packages for CentOS] *****************************
*
skipping: [192.168.56.102]
skipping: [192.168.56.103]
fatal: [192.168.56.109]: FAILED! => {"changed": false, "msg": "Unsupported para
meters for (dnf) module: update_cache Supported parameters include: autoremove,
```



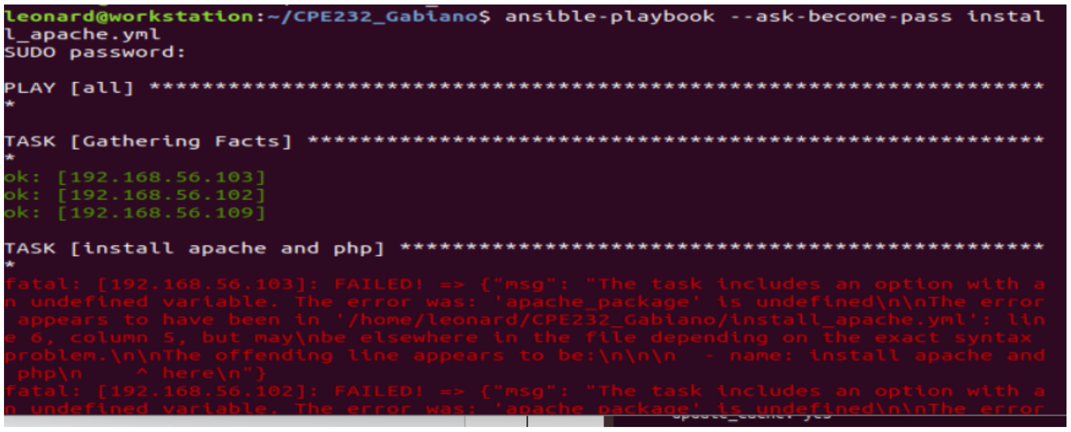
```
skipping: [192.168.56.102]
skipping: [192.168.56.103]
fatal: [192.168.56.109]: FAILED! => {"changed": false, "msg": "Unsupported para
meters for (dnf) module: update_cache Supported parameters include: autoremove,
 conf_file, disable_gpg_check, disablerepo, enablerepo, installroot, list, name
, state"}
        to retry, use: --limit @/home/leonard/CPE232_Gabiano/install_apache.ret
ry

PLAY RECAP *********************************************************************
*
192.168.56.102             : ok=2    changed=0    unreachable=0    failed=0
192.168.56.103             : ok=2    changed=0    unreachable=0    failed=0
192.168.56.109             : ok=1    changed=0    unreachable=0    failed=1

leonard@workstation:~/CPE232_Gabiano$ 
```

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.

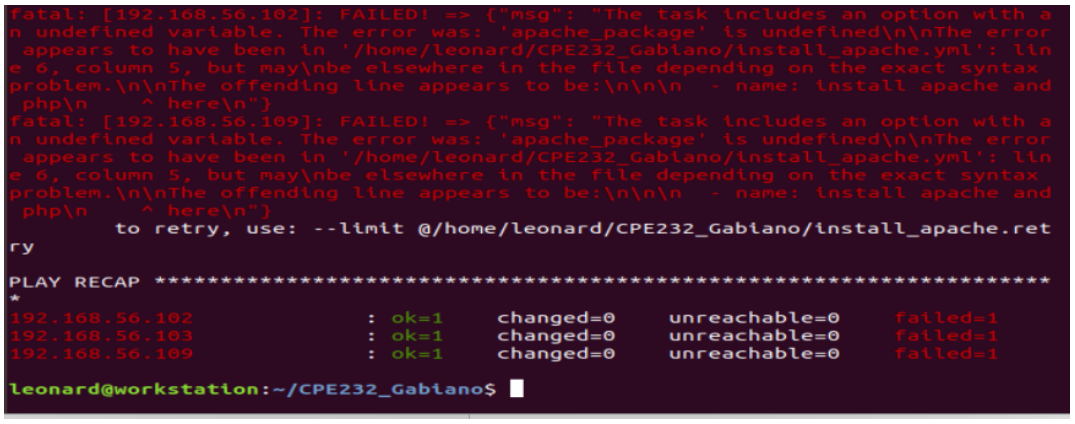


```
---
- hosts: all
  become: true
  tasks:

  - name: install apache and php
    apt:
      name:
        - "{{ apache_package }}"
        - "{{ php_package }}"
      state: latest
      update_cache: yes
```

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

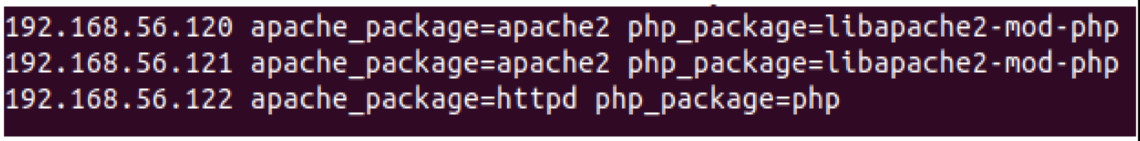


```
leonard@workstation:~/CPE232_Gabiano$ ansible-playbook --ask-become-pass instal
l_apache.yml
SUDO password:

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

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.103]
ok: [192.168.56.102]
ok: [192.168.56.109]

TASK [install apache and php] **************************************************
*
fatal: [192.168.56.103]: FAILED! => {"msg": "The task includes an option with a
n undefined variable. The error was: 'apache_package' is undefined\n\nThe error
 appears to have been in '/home/leonard/CPE232_Gabiano/install_apache.yml': lin
e 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: install apache and
 php\n    ^ here\n"}
fatal: [192.168.56.102]: FAILED! => {"msg": "The task includes an option with a
n undefined variable. The error was: 'apache_package' is undefined\n\nThe error
```



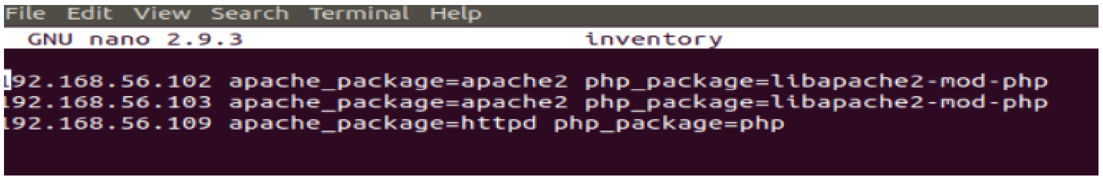
```
fatal: [192.168.56.102]: FAILED! => {"msg": "The task includes an option with a
n undefined variable. The error was: 'apache_package' is undefined\n\nThe error
 appears to have been in '/home/leonard/CPE232_Gabiano/install_apache.yml': lin
e 6, column 5, but may\nbe elsewhere in the file depending on the exact syntax
problem.\n\nThe offending line appears to be:\n\n\n  - name: install apache and
 php\n    ^ here\n"}
fatal: [192.168.56.109]: FAILED! => {"msg": "The task includes an option with a
n undefined variable. The error was: 'apache_package' is undefined\n\nThe error
 appears to have been in '/home/leonard/CPE232_Gabiano/install_apache.yml': lin
e 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: install apache and
 php\n    ^ here\n"}
        to retry, use: --limit @/home/leonard/CPE232_Gabiano/install_apache.ret
ry

PLAY RECAP *********************************************************************
*
192.168.56.102             : ok=1    changed=0    unreachable=0    failed=1
192.168.56.103             : ok=1    changed=0    unreachable=0    failed=1
192.168.56.109             : ok=1    changed=0    unreachable=0    failed=1

leonard@workstation:~/CPE232_Gabiano$
```

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:



```
192.168.56.120 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.121 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.122 apache_package=httpd php_package=php
```

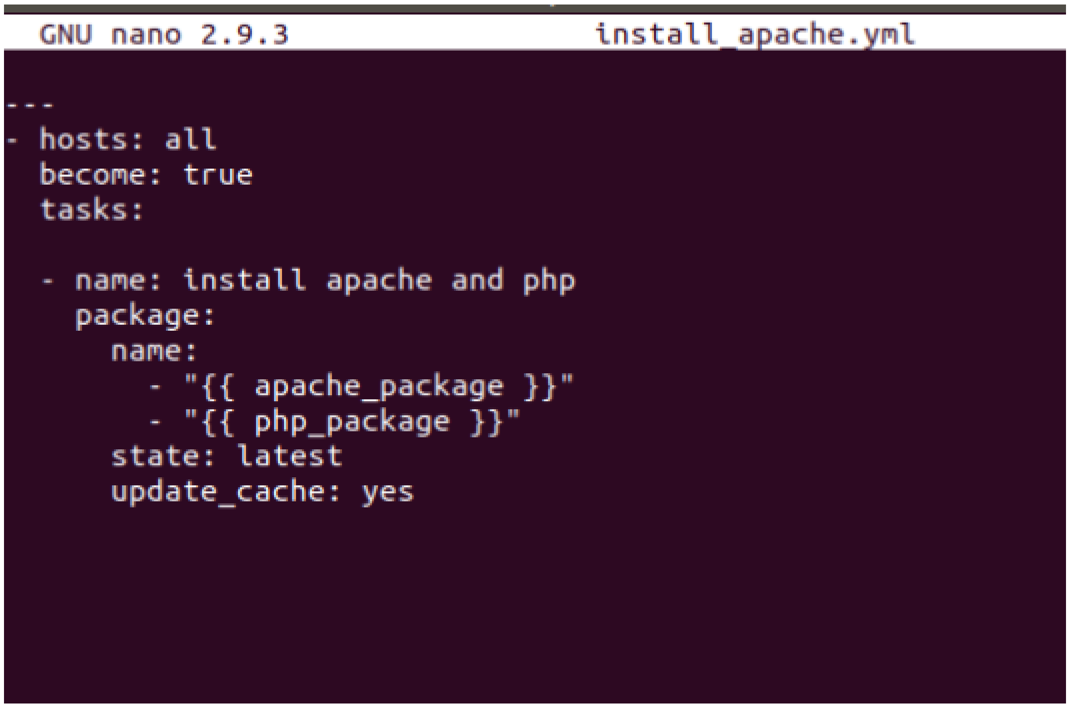
Make sure to save the *inventory* file and exit.



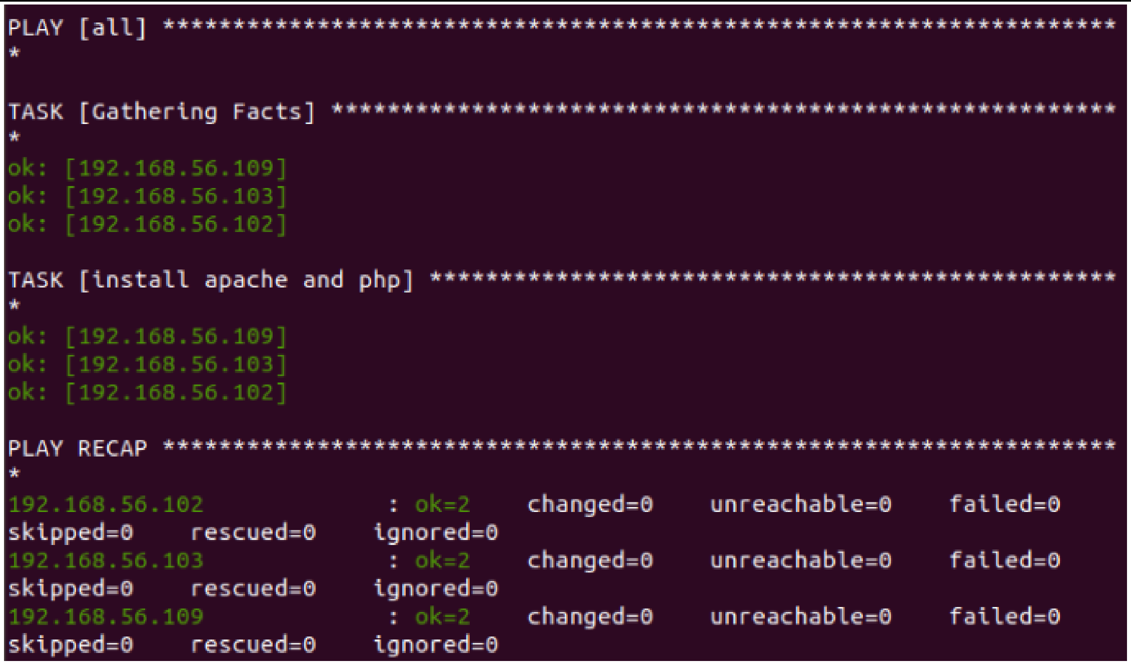
```
  GNU nano 2.9.3                           inventory

192.168.56.102 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.103 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.109 apache_package=httpd php_package=php
```

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.



```
  GNU nano 2.9.3                    install_apache.yml

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

  - name: install apache and php
    package:
      name:
        - "{{ apache_package }}"
        - "{{ php_package }}"
      state: latest
      update_cache: yes
```



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

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]

TASK [install apache and php] **************************************************
*
ok: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]

PLAY RECAP *********************************************************************
*
192.168.56.102             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.109             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Supplementary Activity:

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

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
   Refactoring playbook codes in Linux is essential because it makes them cleaner, more efficient, and easier to maintain. Imagine your playbook is like a recipe for a dish. When you refactor, you're basically cleaning up the steps, removing unnecessary ingredients, and organizing them logically. This simplification reduces errors, speeds up execution, and makes it simpler to add or change tasks in the future. In a nutshell, playbook refactoring is like tidying up your cooking instructions for a smoother and tastier experience.

2. When do we use the “when” command in playbook?

   In Linux, the "when" command is not a standard or widely recognized command. It appears you might be referring to the use of conditionals in Ansible playbooks. In Ansible playbooks, you use the "when" keyword to specify conditions under which a particular task should be executed. For example, you can use "when" to check if a certain file exists or if a service is running before performing an action. This helps ensure that tasks are only carried out when specific conditions are met, making your automation more precise and efficient.

conclusion

In conclusion, the objectives discussed involve using the "when" command in Ansible playbooks to cater to different operating system distributions and applying refactoring techniques to improve playbook code clarity and efficiency. This is particularly useful in environments where multiple Linux distributions are in use, allowing specific tasks to be executed based on the host's distribution. Additionally, the discussion briefly mentions the importance of using Git pull in a collaborative environment to keep code up to date. Lastly, the requirement entails creating a CentOS VM, configuring its network settings, and verifying SSH access, which is essential for practicing and implementing these objectives.