Name: Tendencia, Jasmin Raiza S.	**Date Performed:** 09/18/2023
Course/Section: CPE232 - CPE31S4	**Date Submitted:** 09/19/2023
Instructor: Dr. Jonathan Taylar	**Semester and SY:** 1st/2023-2024

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?

```
tendencia@workstation:~$ cd CPE232_JasminTendencia
tendencia@workstation:~/CPE232_JasminTendencia$ git pull
Already up to date.
tendencia@workstation:~/CPE232_JasminTendencia$
```

Figure 1.1

The displayed output is “Already up to date.” since files in the github was up to date into the Virtual Box knowing that *git pull* allows a server to copy files from the repository, if there are changes from repository then issuing this will update or copy the files in the server.

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."

```
tendencia@workstation:~/CPE232_JasminTendencia$ sudo nano inventory
```

```
  GNU nano 6.2
[ubuntu]
192.168.56.102
192.168.56.103

[centos]
192.168.56.104
```

Figure 2.1. Edited inventory file with IP address of CentOS

```
enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.104  netmask 255.255.255.0  broadcast 192.168.56.255
```

Figure 2.2. IP address of CentOS

```
tendencia@workstation:~/HOA5$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.103]
ok: [192.168.56.102]
ok: [192.168.56.104]

TASK [update repository index] *************************************************
[WARNING]: Updating cache and auto-installing missing dependency: python-apt
fatal: [192.168.56.104]: FAILED! => {"changed": false, "cmd": "apt-get update", "msg": "
[Errno 2] No such file or directory", "rc": 2, "stderr": "", "stderr_lines": [], "stdout
": "", "stdout_lines": []}
changed: [192.168.56.102]
changed: [192.168.56.103]

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

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

PLAY RECAP *********************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0    skipped=0
    rescued=0    ignored=0
192.168.56.103             : ok=4    changed=1    unreachable=0    failed=0    skipped=0
    rescued=0    ignored=0
192.168.56.104             : ok=1    changed=0    unreachable=0    failed=1    skipped=0
    rescued=0    ignored=0

tendencia@workstation:~/HOA5$
```

Figure 2.3. Execution of playbook which resulted to an error

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.

```
 GNU nano 6.2                          install_apache.yml *
---
- 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"
```

Figure 3.1. Edited playbook

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

```
tendencia@workstation:~/HOA5$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.103]
ok: [192.168.56.104]
ok: [192.168.56.102]

TASK [update repository index] *************************************************
skipping: [192.168.56.104]
changed: [192.168.56.103]
changed: [192.168.56.102]

TASK [install apache2 package] *************************************************
skipping: [192.168.56.104]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [add PHP support for apache] **********************************************
skipping: [192.168.56.104]
ok: [192.168.56.103]
ok: [192.168.56.102]

PLAY RECAP *********************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.104             : ok=1    changed=0    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
```

Figure 3.2. Output after running the command which is successfully executed in the IP address of Server 1, Server 2, and CentOS 7

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.

```
 GNU nano 6.2                                                  install_ap
---
- hosts: all
  become: true
  tasks:

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

  - name: install apache2 package
    apt:
      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"
```

Figure 4.1. Edited playbook

Note: I change the "CentOS" to == ""Debian"

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

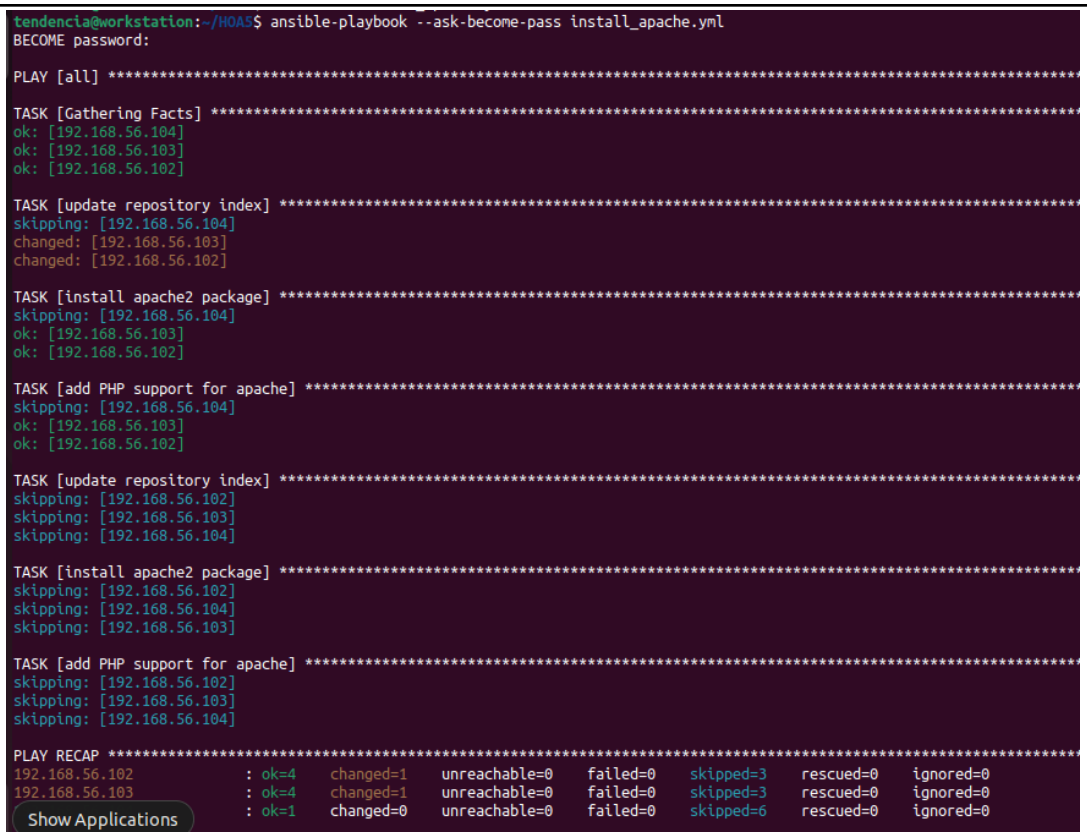
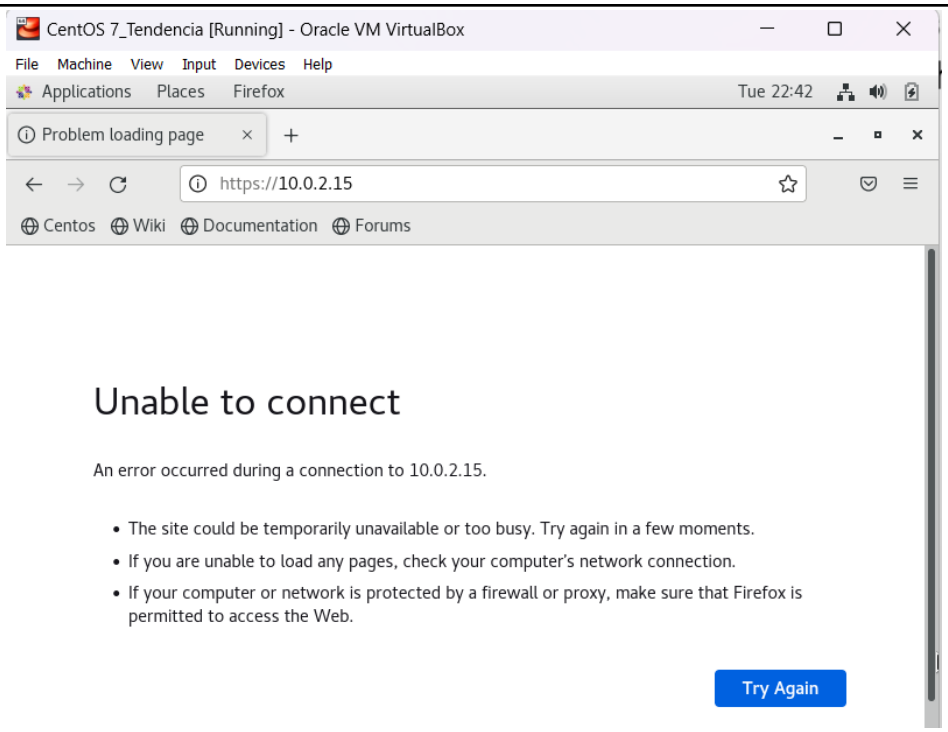


Figure 4.2. Running the command, successful run for Server 1, Server 2, and CentOS 7

It is noticeable here that the CentOS skipped 3 in the playbook since those three were meant for Ubuntu.

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.

5.2 Issue the following command to start the service:

sudo systemctl start httpd

(When prompted, enter the sudo password)

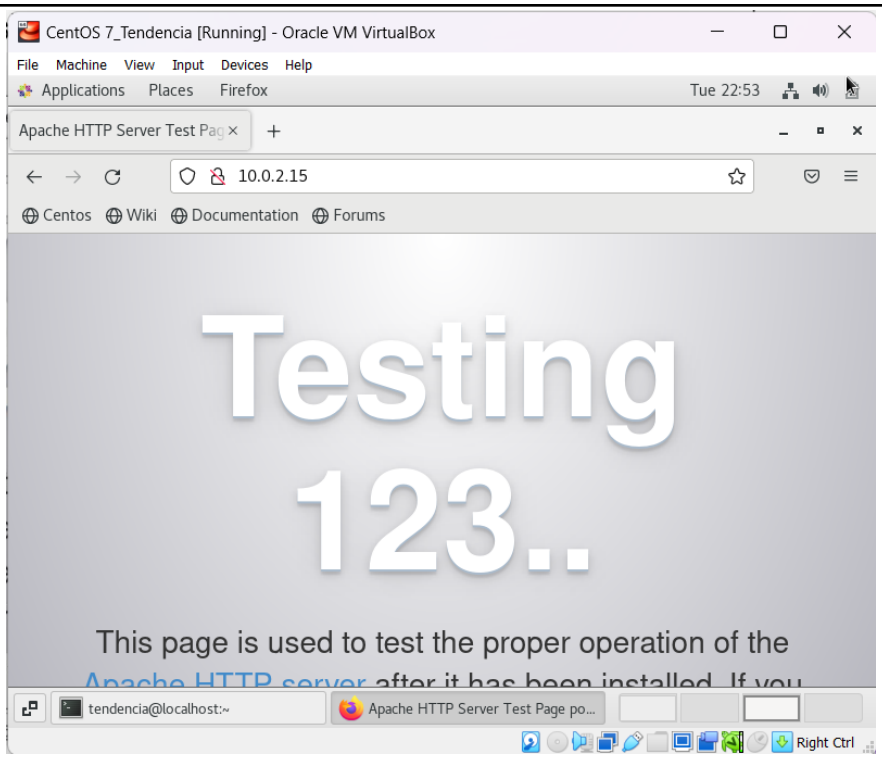
sudo firewall-cmd --add-port=80/tcp

(The result should be a success)

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

I received an error here at first since it is not yet installed so I tried the command: *sudo yum install httpd.* Here, after upgrading the systems the terminal allowed the user to change the the status of the firewall of centOS.

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)



Yes, since the firewall has been activated as well as the Apache HTTP server.

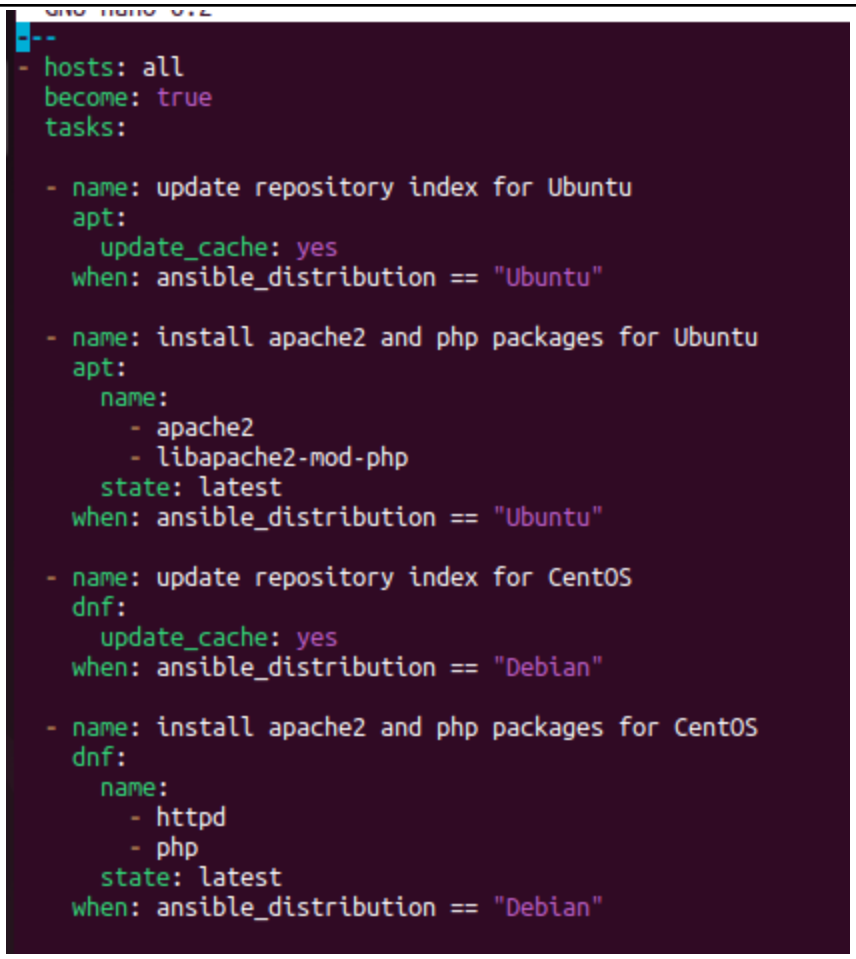
Task 2: Refactoring playbook

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.



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

  - name: update repository index for 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 == "Debian"

  - name: install apache2 and php packages for CentOS
    dnf:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution == "Debian"
```

Figure 1.1

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

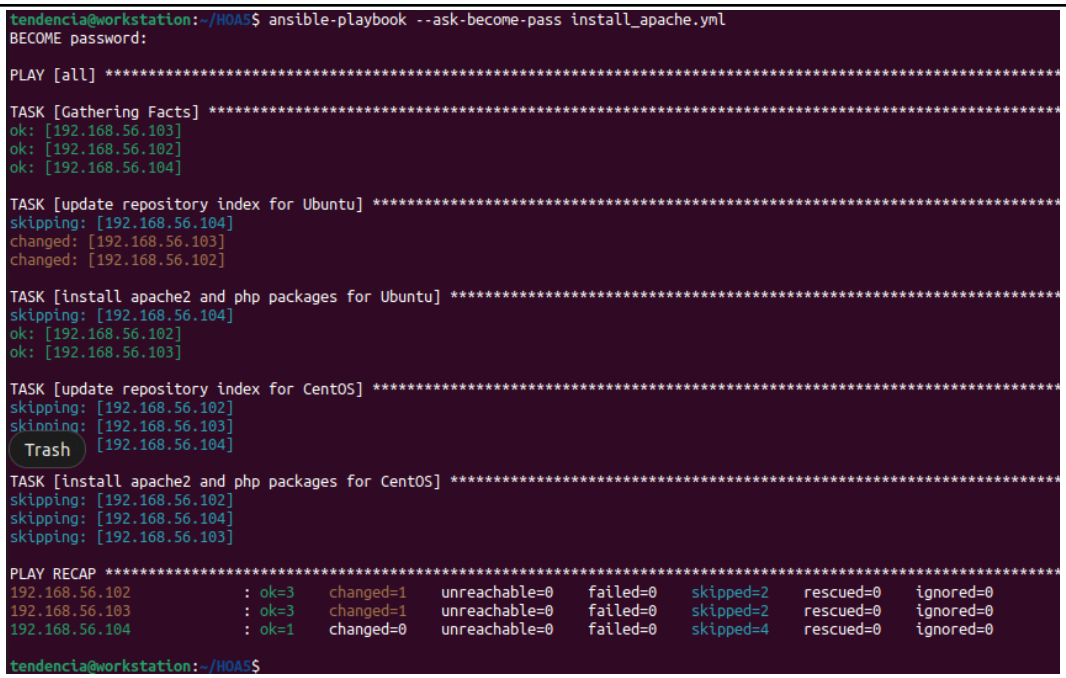


Figure 1.2.

Once again, activating the updated 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:

```
---
- 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.

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

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

- name: install apache2 and php packages for CentOS
  dnf:
    name:
      - httpd
      - php
    state: latest
    update_cache: yes
  when: ansible_distribution == "Debian"
```

Thunderbird Mail

Figure 2.1

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

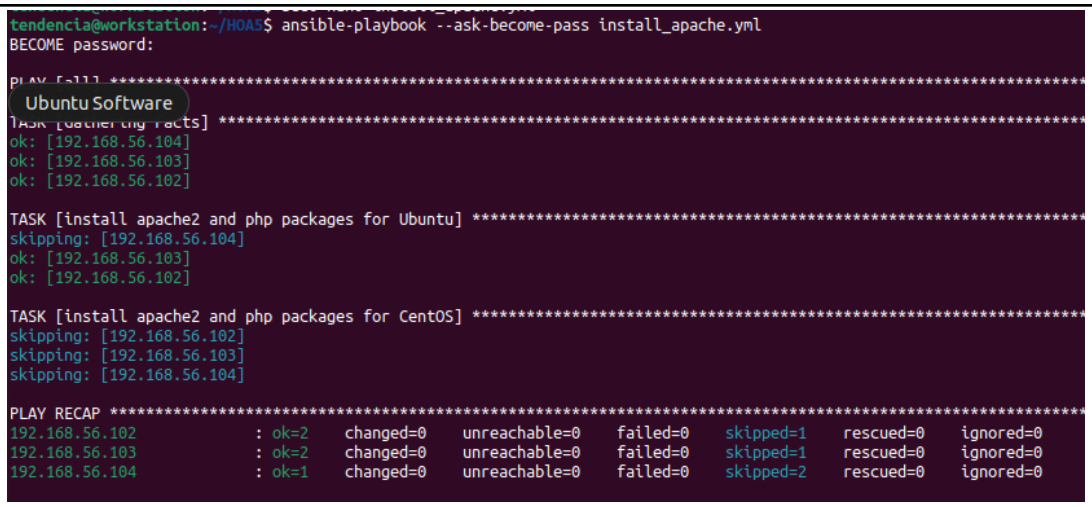
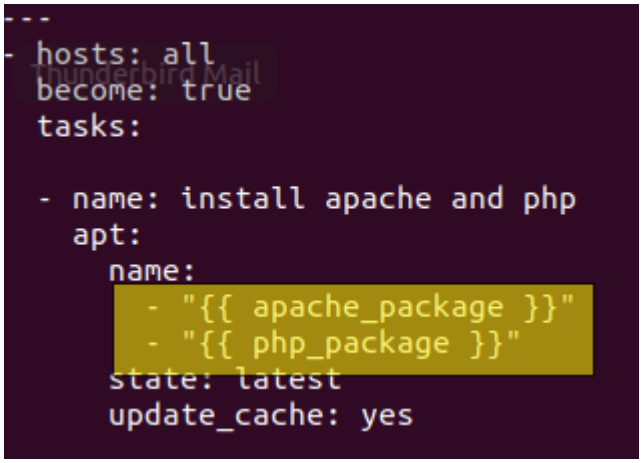


Figure 2.2

Here, another activation of the updated playbook.

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 6.2                                    install_apache.yml
---
- 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.

```
tendencia@workstation:~/HOA5$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *****************************************************************************************************
ok: [192.168.56.103]
ok: [192.168.56.102]
ok: [192.168.56.104]

TASK [install apache and php] **********************************************************************************************
fatal: [192.168.56.102]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'apache_
efined\n\nThe error appears to be in '/home/tendencia/HOA5/install_apache.yml': line 6, column 5, but may\nbe elsewhere in t
\n\nThe offending line appears to be:\n\n\n  - name: install apache and php\n    ^ here\n"}
fatal: [192.168.56.103]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'apache_
efined\n\nThe error appears to be in '/home/tendencia/HOA5/install_apache.yml': line 6, column 5, but may\nbe elsewhere in t
\n\nThe offending line appears to be:\n\n\n  - name: install apache and php\n    ^ here\n"}
fatal: [192.168.56.104]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'apache_
efined\n\nThe error appears to be in '/home/tendencia/HOA5/install_apache.yml': line 6, column 5, but may\nbe elsewhere in t
\n\nThe offending line appears to be:\n\n\n  - name: install apache and php\n    ^ here\n"}

PLAY RECAP *****************************************************************************************************************
192.168.56.102             : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
192.168.56.104             : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
```

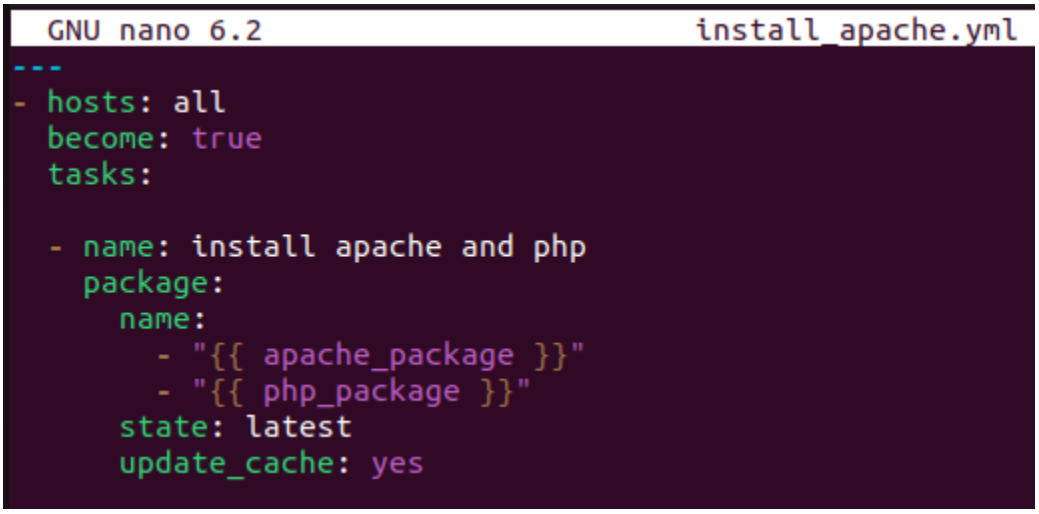
4. Unfortunately, task 2.3 was not successful. It's because we need to change something in the inventory file so that the variables we declared will be in place. Edit the *inventory* file and follow the below configuration:

```
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.

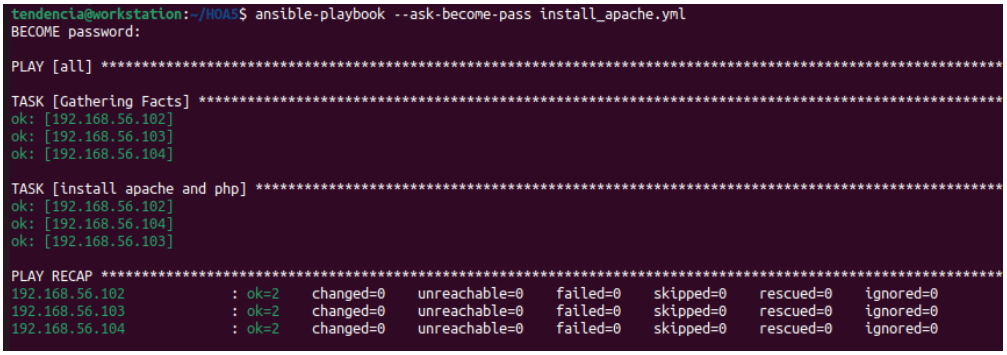
```
[ubuntu]
192.168.56.102 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.103 apache_package=apache2 php_package=libapache2-mod-php

[centos]
192.168.56.104 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



Changing apt to package

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



Executing the playbook resulted into a successful output

Supplementary Activity:

1. Create a playbook that could do the previous tasks in Red Hat OS.
    - Same with CentOS output.

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?

   Playbook code refactoring holds significance due to its capacity to be comprehended easily and in adding new features. This process simplifies supplementary code without altering the program's output or behavior.

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

   In a playbook, the "when" command is employed when multiple conditions need to be integrated into the process.

Conclusion:

Throughout this activity, I have learned that working with Linux-based Ubuntu and CentOS 7 systems revolves around the strategic utilization of the “when” command in playbooks and the implementation of effective code refactoring techniques are pivotal. Moreover, the practice of code refactoring can be considered as a skill since it ensures that the playbook codes remain comprehensible and adaptable, thus simplifying the incorporation of new features without altering the program’s output or behavior. Furthermore, with these, it ultimately contributed to more efficient and reliable automation solutions.