

Name: Peñas, Issa Victoria H.	Date Performed: 09/13/2022
Course/Section: CPE 232 - CPE31S22	Date Submitted: 09/15/2022
Instructor: Dr. Jonathan Taylar	Semester and SY: 1st Semester (SY: 2022 - 2023)
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: <p>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.</p> <p>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.</p> <p>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.</p>	
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? <ul style="list-style-type: none"> - The displayed output was 'Already up to date.' in which means that the files from the github was up to date into the Virtual Box, as the command 'git pull' allows a server to copy the files from the connected repository, if there are changes from the repository then issuing the command will update or copy the files in the server. 	

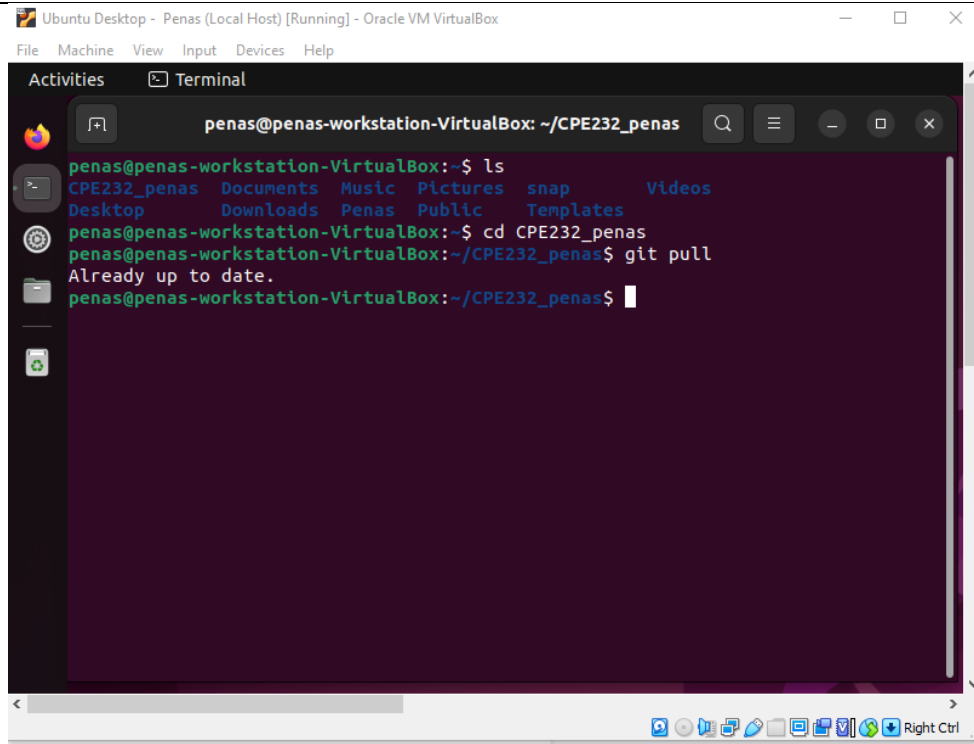


Figure 1.1. Shows the output after pulling the files from GitHub

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."

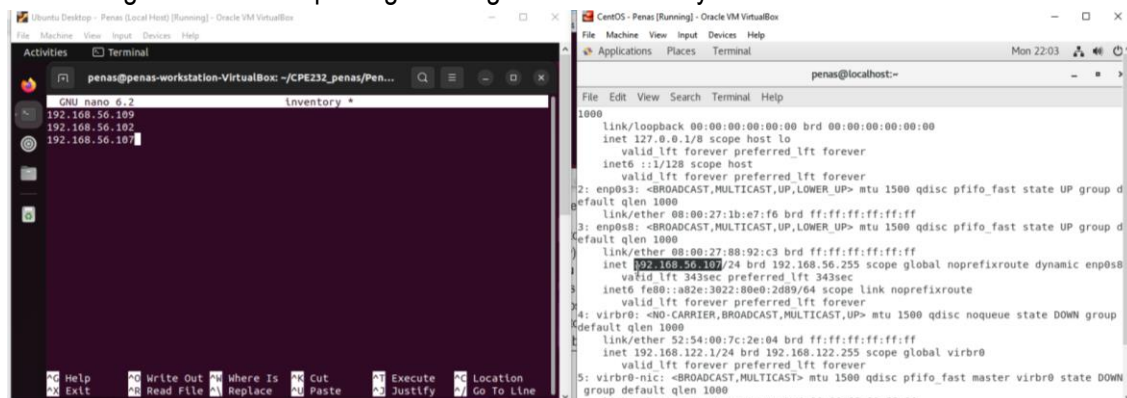


Figure 1.2. Showing the IP Address of CentOs inputting in the ansible inventory

```

penas@penas-workstation-VirtualBox:~/CPE232_penas/penas/ansible$ ansible-playbook
k --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
The authenticity of host '192.168.56.107 (192.168.56.107)' can't be established.
ED25519 key fingerprint is SHA256:ZdxkCYnrVrkrQL3qSY1ccgnyVw7ToIlQGdTPBXip2W0.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
fatal: [192.168.56.107]: UNREACHABLE! => {"changed": false, "msg": "Failed to co
nnect to the host via ssh: Warning: Permanently added '192.168.56.107' (ED25519)
to the list of known hosts.\r\nLoad key \"/home/penas/.ssh/": Is a directory\r\
npenas@192.168.56.107: Permission denied (publickey,gssapi-keyex,gssapi-with-mlc
,password).", "unreachable": true}
fatal: [192.168.56.109]: UNREACHABLE! => {"changed": false, "msg": "Failed to co
nnect to the host via ssh: ssh: connect to host 192.168.56.109 port 22: No route
to host", "unreachable": true}
fatal: [192.168.56.102]: UNREACHABLE! => {"changed": false, "msg": "Failed to co
nnect to the host via ssh: ssh: connect to host 192.168.56.102 port 22: No route
to host", "unreachable": true}

PLAY RECAP *****
192.168.56.102      : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.107      : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.109      : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0

```

Figure 1.3. Executing the playbook which outputted 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.
---
- 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 1.4. Editing the playbook

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

Note: The PC cannot open 4 Virtual PC (Server 1, Server 2, CentOS, Local Host) simultaneously resulting in the PC freezing, please bear with me

```
penas@penas-workstation-VirtualBox: ~/CPE232_penas/...
penas@penas-workstation-VirtualBox:~/CPE232_penas/penas/ansible$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.107]
fatal: [192.168.56.102]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.102 port 22: No route to host", "unreachable": true}
fatal: [192.168.56.109]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.109 port 22: No route to host", "unreachable": true}

TASK [update repository index] *****
skipping: [192.168.56.107]

TASK [install apache2 package] *****
skipping: [192.168.56.107]

TASK [add PHP support for apache] *****
skipping: [192.168.56.107]

PLAY RECAP *****
192.168.56.102      : ok=0    changed=0    unreachable=1    failed=0
skipped=0          rescued=0    ignored=0
192.168.56.107      : ok=1    changed=0    unreachable=0    failed=0
skipped=3          rescued=0    ignored=0
192.168.56.109      : ok=0    changed=0    unreachable=1    failed=0
skipped=0          rescued=0    ignored=0
penas@penas-workstation-VirtualBox:~/CPE232_penas/penas/ansible$
```

Figure 1.5.1 The output after running the code which is successfully executed in the IP Address of CentOS

```
penas@penas-workstation-VirtualBox: ~/CPE232_penas/penas/...
penas@penas-workstation-VirtualBox:~/CPE232_penas/penas/ansible$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
fatal: [192.168.56.102]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.102 port 22: No route to host", "unreachable": true}
fatal: [192.168.56.107]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.107 port 22: No route to host", "unreachable": true}
ok: [192.168.56.109]

TASK [update repository index] *****
changed: [192.168.56.109]

TASK [install apache2 package] *****
ok: [192.168.56.109]

TASK [add PHP support for apache] *****
ok: [192.168.56.109]

PLAY RECAP *****
192.168.56.102      : ok=0    changed=0    unreachable=1    failed=0    s
skipped=0          rescued=0    ignored=0
192.168.56.107      : ok=0    changed=0    unreachable=1    failed=0    s
skipped=0          rescued=0    ignored=0
192.168.56.109      : ok=4    changed=1    unreachable=0    failed=0    s
skipped=0          rescued=0    ignored=0
```

Figure 1.5.2. The output after running the code which is successfully executed in the IP Address of Server 1

```

penas@penas-workstation-VirtualBox: ~/CPE232_penas/Pen...
kipped=0    rescued=0    ignored=0

penas@penas-workstation-VirtualBox:~/CPE232_penas/Penas/ansible$ ansible-playbook
k --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.102]
fatal: [192.168.56.109]: UNREACHABLE! => {"changed": false, "msg": "Failed to co
nnect to the host via ssh: ssh: connect to host 192.168.56.109 port 22: No route
to host", "unreachable": true}
fatal: [192.168.56.107]: UNREACHABLE! => {"changed": false, "msg": "Failed to co
nnect to the host via ssh: ssh: connect to host 192.168.56.107 port 22: No route
to host", "unreachable": true}

TASK [update repository index] *****
changed: [192.168.56.102]

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

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

PLAY RECAP *****
192.168.56.102      : ok=4    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.107      : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.109      : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0

```

Figure 1.5.3. The output after running the code which is successfully executed in the IP Address of Server 2

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
        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.

```

GNU nano 6.2      install_apache.yml
- 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
    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 1.6. Editing the Playbook

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

- In Figure 1.6. It shows that the IP Address of CentOS which is 192.168.56.107 skipped 3 command codes from the playbook since those 3 were meant for the Ubuntu, vice versa

```

PLAY [all] *****
TASK [Gathering Facts] *****
ok: [192.168.56.102]
ok: [192.168.56.109]
ok: [192.168.56.107]

TASK [update repository index] *****
skipping: [192.168.56.107]
changed: [192.168.56.109]
changed: [192.168.56.102]

TASK [install apache2 package] *****
skipping: [192.168.56.107]
ok: [192.168.56.109]
ok: [192.168.56.102]

TASK [add PHP support for apache] *****
skipping: [192.168.56.107]
ok: [192.168.56.109]
ok: [192.168.56.102]

TASK [update repository index] *****
skipping: [192.168.56.109]
skipping: [192.168.56.102]
ok: [192.168.56.107]

TASK [install apache2 package] *****
skipping: [192.168.56.109]
skipping: [192.168.56.102]
changed: [192.168.56.107]

TASK [add PHP support for apache] *****
skipping: [192.168.56.109]
skipping: [192.168.56.102]
changed: [192.168.56.107]

PLAY RECAP *****
192.168.56.102 : ok=4   changed=1   unreachable=0   failed=0   skipped=3
               rescued=0   ignored=0
192.168.56.107 : ok=4   changed=2   unreachable=0   failed=0   skipped=3
               rescued=0   ignored=0
192.168.56.109 : ok=4   changed=1   unreachable=0   failed=0   skipped=3
               rescued=0   ignored=0

```

Figure 1.7. The output after running the code is successfully executed in the IP Address of CentOS, Server 1 and 2

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.

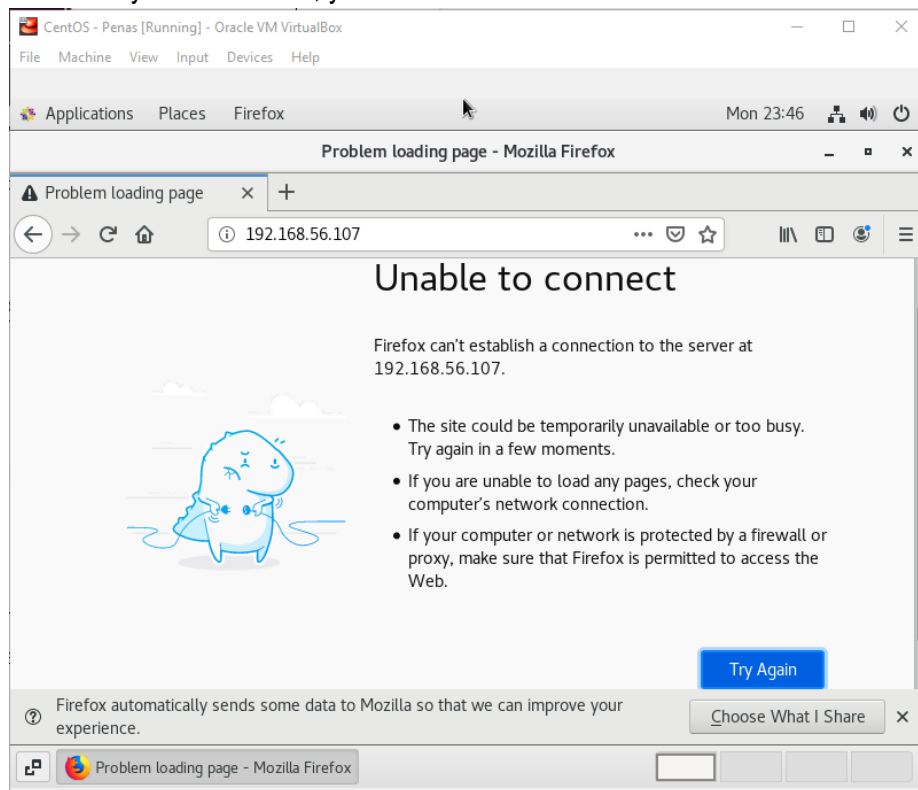


Figure 1.8. The output shows that the admin was unable to connect to the Apache HTTP Server

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)

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

Figure 1.9. After upgrading all the systems the terminal allowed the user to change 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)

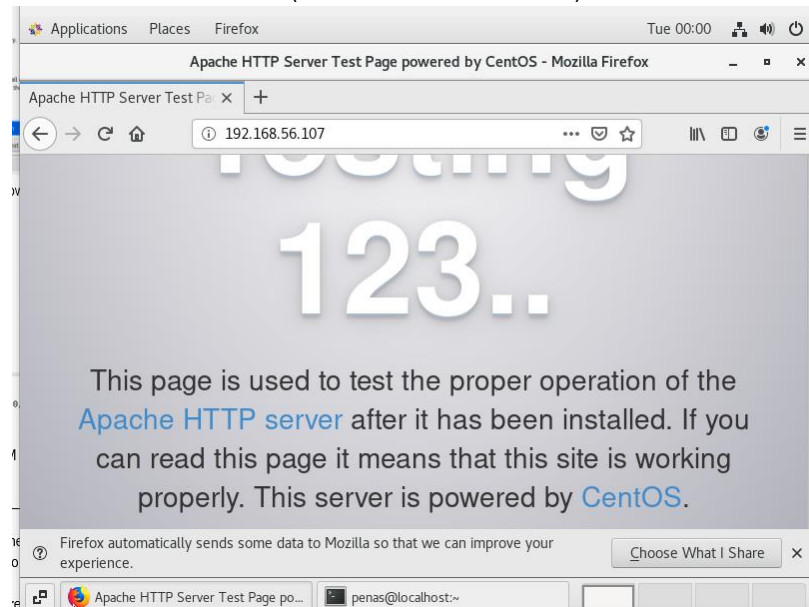


Figure 1.10. After activating the firewall and the Apache HTTP server

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 make Ansible run 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.

```
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 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"
```

Figure 1.11. Editing the said playbook

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

```
penas@penas-workstation-VirtualBox:~/CPE232_penas/penas/ansible$ ansible-playbook --ask-
become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.107]
ok: [192.168.56.109]
ok: [192.168.56.102]

TASK [update repository index] *****
skipping: [192.168.56.107]
changed: [192.168.56.109]
changed: [192.168.56.102]

TASK [install apache2 package and php packages for Ubuntu] *****
skipping: [192.168.56.107]
ok: [192.168.56.109]
ok: [192.168.56.102]

TASK [update repository index for CentOS] *****
skipping: [192.168.56.109]
skipping: [192.168.56.102]
ok: [192.168.56.107]

TASK [install apache and php packages for CentOS] *****
skipping: [192.168.56.109]
skipping: [192.168.56.102]
ok: [192.168.56.107]

PLAY RECAP *****
192.168.56.102      : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.107      : ok=3    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.109      : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

Figure 1.12. Activating the updated playbook once again

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:

```

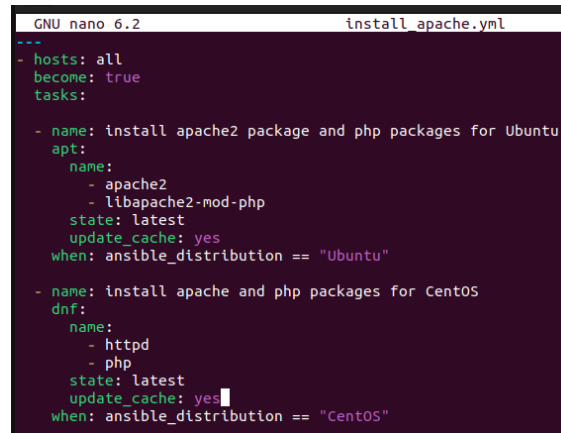
---
- hosts: all
  become: true
  tasks:

    - 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.



```

GNU nano 6.2      install_apache.yml
---
- hosts: all
  become: true
  tasks:

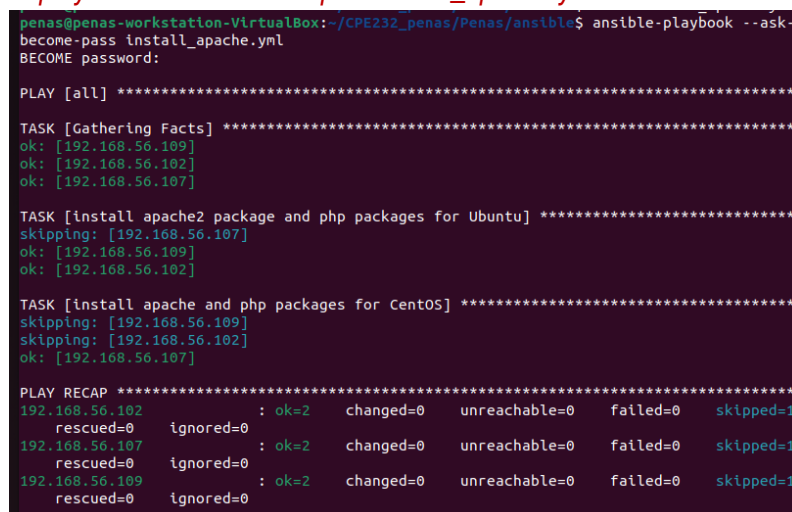
    - name: install apache2 package and php packages for Ubuntu
      apt:
        name:
          - apache2
          - libapache2-mod-php
        state: latest
        update_cache: yes
      when: ansible_distribution == "Ubuntu"

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

```

Figure 1.13. Editing the said playbook and summarizing commands

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



```

penas@penas-workstation-VirtualBox:~/CPE232_penas/Penas/ansible$ ansible-playbook --ask-
become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.107]

TASK [install apache2 package and php packages for Ubuntu] *****
skipping: [192.168.56.107]
ok: [192.168.56.109]
ok: [192.168.56.102]

TASK [install apache and php packages for CentOS] *****
skipping: [192.168.56.109]
skipping: [192.168.56.102]
ok: [192.168.56.107]

PLAY RECAP *****
192.168.56.102 : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.107 : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.109 : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0

```

Figure 1.14. Activating the updated playbook once again

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.

```
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
```

Figure 1.15. Editing the said playbook and summarizing the commands

```
PLAY RECAP *****
192.168.56.102 : ok=1 changed=0 unreachable=0 failed=1 skipped=0 rescued=0
192.168.56.107 : ok=1 changed=0 unreachable=0 failed=1 skipped=0 rescued=0
192.168.56.109 : ok=1 changed=0 unreachable=0 failed=1 skipped=0 rescued=0

penas@penas-workstation-VirtualBox:~/CPE222_penas/penas/ansible$ nano install_apache.yml
penas@penas-workstation-VirtualBox:~/CPE222_penas/penas/ansible$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.107]
ok: [192.168.56.109]
ok: [192.168.56.102]

TASK [install apache and php] *****
[task] [192.168.56.109]: FAILED! => [msg]: "The task includes an option with an undefined variable. The error was: 'apache_package' is undefined\n\nthe error appears to be in /home/penas/CPE222_penas/penas/ansible/install_apache.yml: line 8, column 5, but may be elsewhere in the file depending on the exact syntax problem.\n\nthe offending line appears to be:\n\n  - name: install apache and php\n    ^ here\n"
fatal: [192.168.56.102]: FAILED! => [msg]: "The task includes an option with an undefined variable. The error was: 'apache_package' is undefined\n\nthe error appears to be in /home/penas/CPE222_penas/penas/ansible/install_apache.yml: line 8, column 5, but may be elsewhere in the file depending on the exact syntax problem.\n\nthe offending line appears to be:\n\n  - name: install apache and php\n    ^ here\n"
fatal: [192.168.56.107]: FAILED! => [msg]: "The task includes an option with an undefined variable. The error was: 'apache_package' is undefined\n\nthe error appears to be in /home/penas/CPE222_penas/penas/ansible/install_apache.yml: line 8, column 5, but may be elsewhere in the file depending on the exact syntax problem.\n\nthe offending line appears to be:\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
192.168.56.107 : ok=1 changed=0 unreachable=0 failed=1 skipped=0 rescued=0
192.168.56.109 : ok=1 changed=0 unreachable=0 failed=1 skipped=0 rescued=0
```

Figure 1.16. Executing the summarized playbook again which resulted in failing on installing Apache

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 6.2 inventory
192.168.56.109 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.102 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.107 apache_package=httpd php_package=php
```

Figure 1.17. Editing the *inventory* file by appending some commands

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](https://docs.ansible.com/ansible/latest/modules/package_module.html)

```
penas@penas-workstation-VirtualBox:~/CPE232_penas/Penas/ansible$ ansible-playbook --ask-
become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.107]
ok: [192.168.56.102]

TASK [install apache and php] *****
[WARNING]: Updating cache and auto-installing missing dependency: python-apt
fatal: [192.168.56.107]: FAILED! => {"changed": false, "cmd": "apt-get update", "msg": "[Errno 2] No such file or directory", "rc": 2}
ok: [192.168.56.109]
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
192.168.56.107 : ok=1 changed=0 unreachable=0 failed=1 skipped=0
rescued=0 ignored=0
192.168.56.109 : ok=2 changed=0 unreachable=0 failed=0 skipped=0
rescued=0 ignored=0
```

Figure 1.18. The output when the *apt* into a *package* where its output resulted on error

```

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

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

```

Figure 1.19. Replacing the *apt* to *package* in the playbook

```

penas@penas-workstation-VirtualBox:~/CPE232_penas/Penas/ansible$ ansible-playbook --ask
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.107]
ok: [192.168.56.102]

TASK [install apache and php] *****
ok: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.107]

PLAY RECAP *****
192.168.56.102      : ok=2    changed=0    unreachable=0    failed=0    skipped=0
192.168.56.107      : ok=2    changed=0    unreachable=0    failed=0    skipped=0
192.168.56.109      : ok=2    changed=0    unreachable=0    failed=0    skipped=0

```

Figure 1.20. After replacing the commands, executing the playbook resulted in a successful output

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

- Executing the command where the playbook was already summarized into 1 sector for both CentOS and Ubuntu resulted in a successful output that took minimum time to display the output

Reflections:

Answer the following:

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

- In the previous steps in this procedure the student summarized the command from 6 sectors where 3 of those commands were for CentOS and Ubuntu making it redundant, as those commands were the same yet on different distributions, executing it will take much longer as it runs the commands from the playbook 6 times compared to the summarized playbook which is in **Figure 1.19.** and **Figure 1.17.** which has 1 sector of command for the two said distributions, executing it will take a bit of time yet not as long as the 6 sectors of command. The System Administrator can efficiently update and install packages or modules on different servers whenever refactoring the playbook.

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

- Based on the activity, the “when” command can also be used to distinguish which executable commands are for the described OS, where it correlates to the “If” statement that if certain parameters have been met then the command will be executed, otherwise it'll be skipped or ignored by the system, same as the when command yet it makes the playbook lengthy. In order to solve the time and complexity issue, Refactoring commands is necessary where the admin shortens out the commands inside the playbook by narrowing it down into packages, although it fails at first because the “inventory” text file is lacking certain commands that will make it suitable for package installation. Once finished editing and appending the selected files, it'll display a successful execution of commands and it doesn't skip any OS since the package is meant for all OS connected to the local machine. Distinguishing the commands to be executed, while also creating shortcuts for the said commands while still retaining their original purpose. First, we've made two separate commands for Ubuntu and CentOS that allows the admin to install apache and add PHP for the respective OS. Since the two separate commands are all compiled on a single playbook, the system is having a hard time figuring out which commands are going to be used on a specific OS, which usually takes time to process the whole update for the connected servers.