Name: Aaron Martin P. Caro	**Date Performed: 19/09/2023**
Course/Section: CPE31S5	**Date Submitted: 26/09/2023**
Instructor: Sir. Roman Richard	**Semester and SY: 1st sem 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.

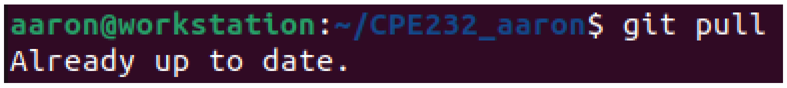
It is a best practice in ansible when you are working in a collaborative environment to use the command git pull. git pull is a Git command used to update the local version of a repository from a remote. By default, git pull does two things. Updates the current local working branch (currently checked 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

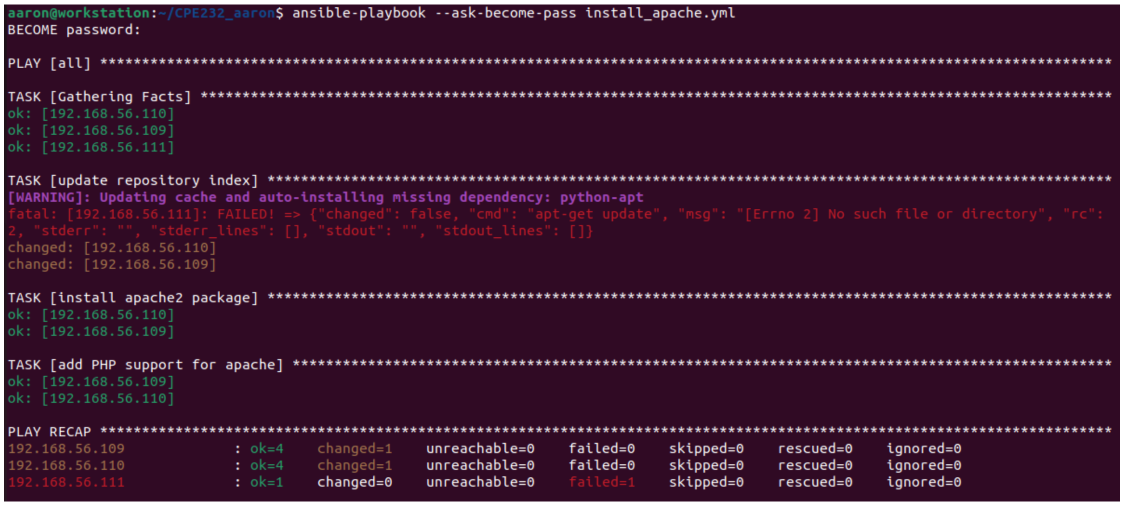
1. In the local machine, make sure you are in the local repository directory (*CPE232_yourname*). Issue the command git pull. When prompted, enter the correct passphrase or password. Describe what happened when you issue this command. Did something happen? Why?



```
aaron@workstation:~/CPE232_aaron$ git pull
Already up to date.
```

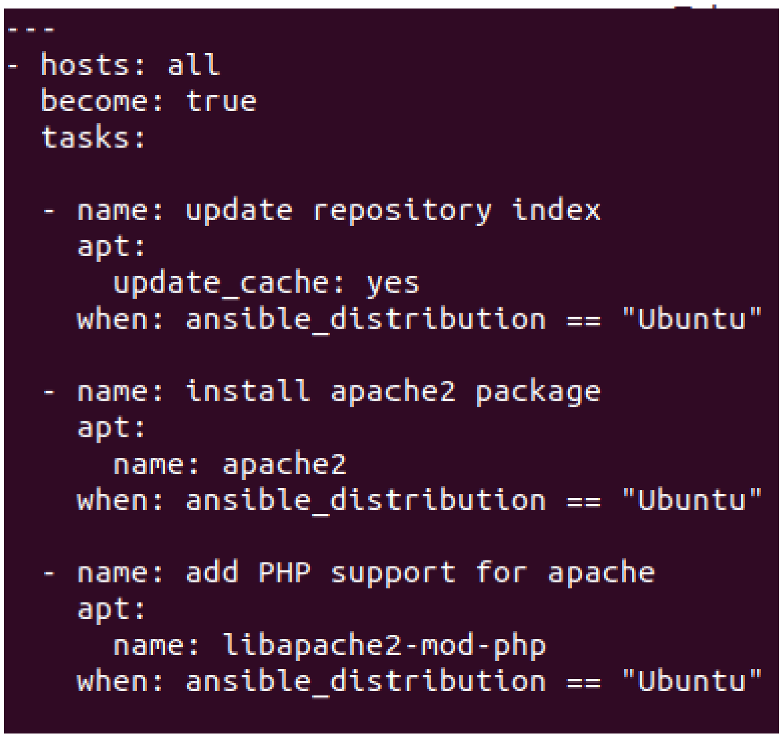
It basically brings in the repository as it is in the remote repository but since my repository is up to date it did not do anything.

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."



```
aaron@workstation:~/CPE232_aaron$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ************************************************************************************************
ok: [192.168.56.110]
ok: [192.168.56.109]
ok: [192.168.56.111]

TASK [update repository index] ****************************************************************************************
[WARNING]: Updating cache and auto-installing missing dependency: python-apt
fatal: [192.168.56.111]: FAILED! => {"changed": false, "cmd": "apt-get update", "msg": "[Errno 2] No such file or directory", "rc":
2, "stderr": "", "stderr_lines": [], "stdout": "", "stdout_lines": []}
changed: [192.168.56.110]
changed: [192.168.56.109]

TASK [install apache2 package] ****************************************************************************************
ok: [192.168.56.110]
ok: [192.168.56.109]

TASK [add PHP support for apache] *************************************************************************************
ok: [192.168.56.109]
ok: [192.168.56.110]

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

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"
```

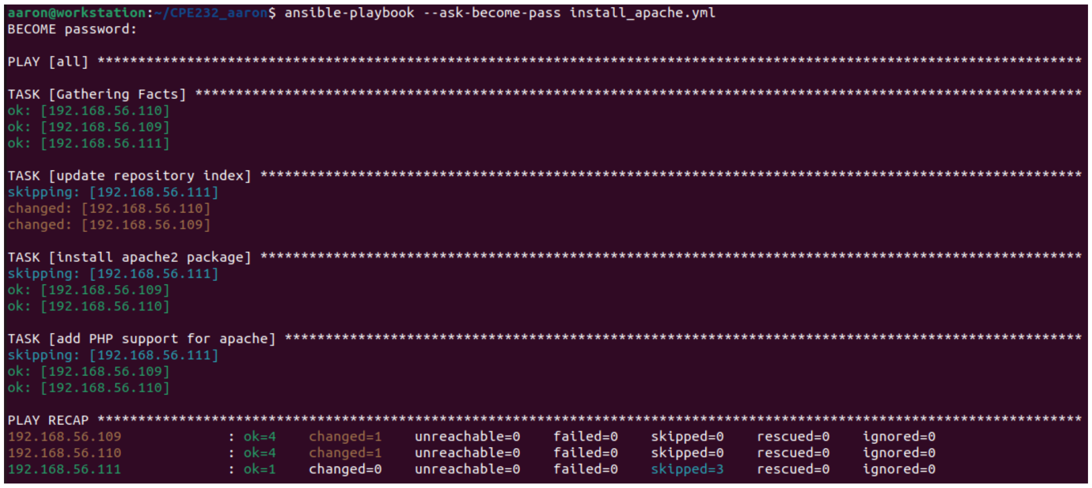
Make sure to save the file and exit.

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

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]
```

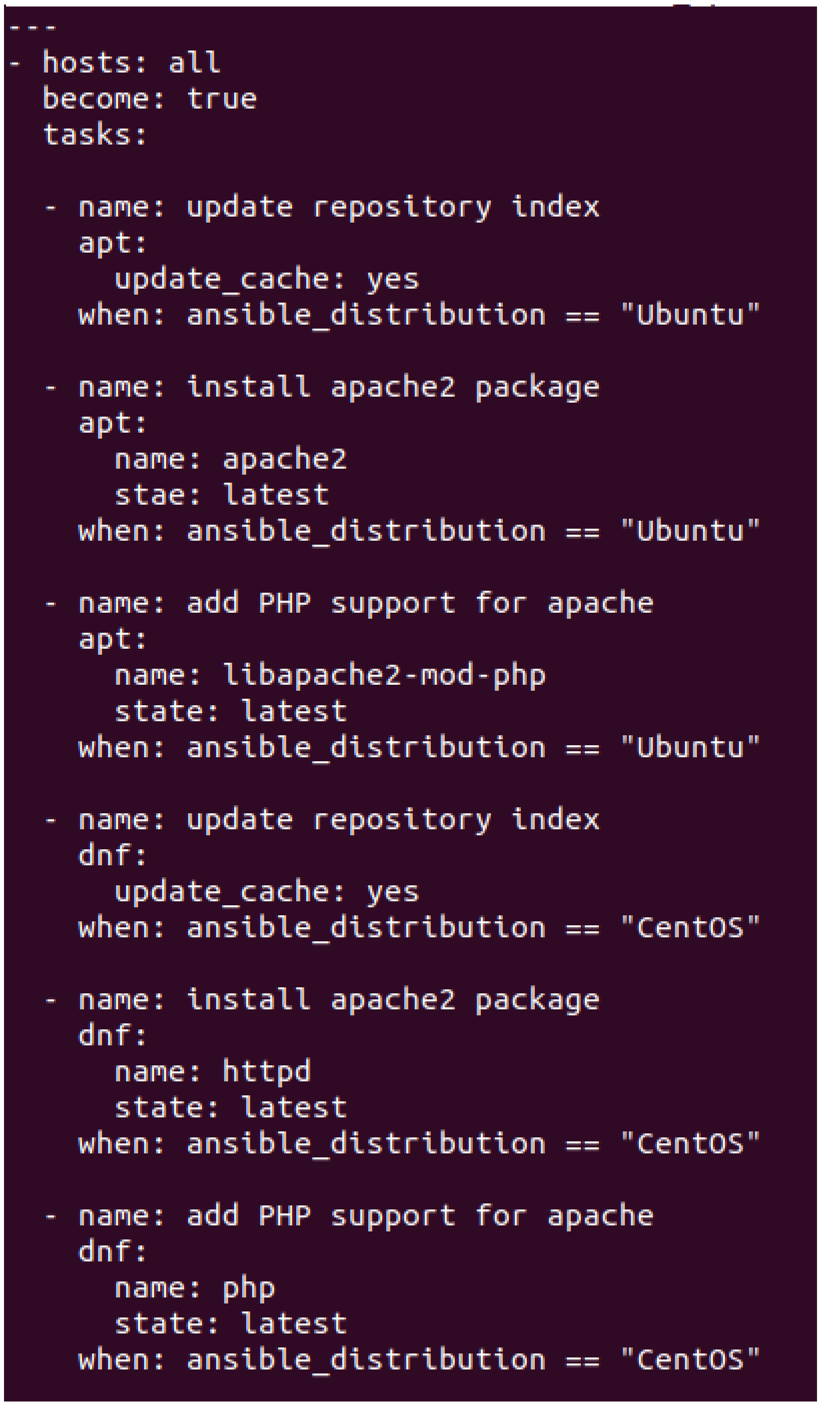
Note: This will work also if you try. Notice the changes are highlighted.



```
aaron@workstation:~/CPE232_aaron$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *************************************************************************************************
ok: [192.168.56.110]
ok: [192.168.56.109]
ok: [192.168.56.111]

TASK [update repository index] *****************************************************************************************
skipping: [192.168.56.111]
changed: [192.168.56.110]
changed: [192.168.56.109]

TASK [install apache2 package] *****************************************************************************************
skipping: [192.168.56.111]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [add PHP support for apache] **************************************************************************************
skipping: [192.168.56.111]
ok: [192.168.56.109]
ok: [192.168.56.110]

PLAY RECAP *************************************************************************************************************
192.168.56.109             : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.110             : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.111             : ok=1    changed=0    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
```

It resulted in skipping the centos vm since it did not match the Ubuntu specification

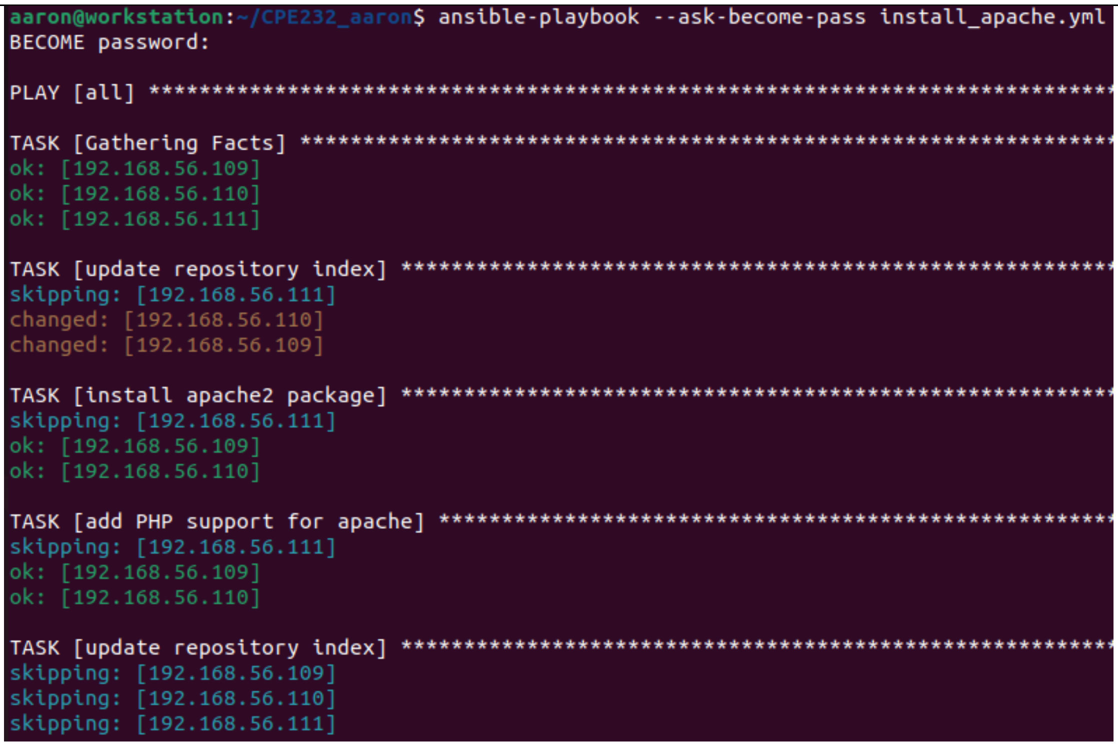
4. Edit the *install_apache.yml* file and insert the lines shown below.



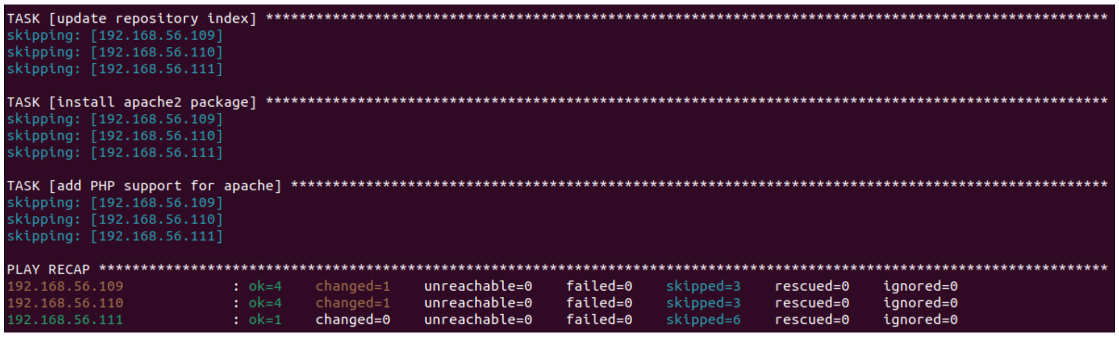
```
---
- hosts: all
  become: true
  tasks:

  - 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.



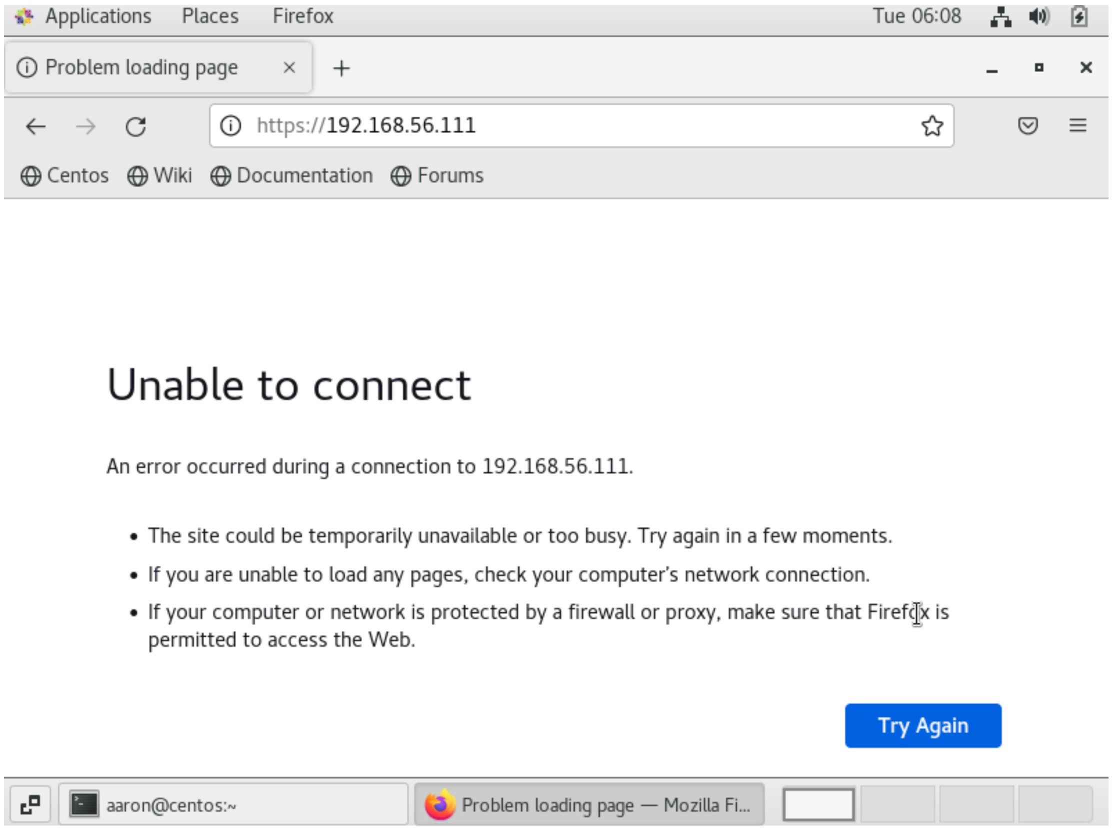
```
aaron@workstation:~/CPE232_aaron$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************************
ok: [192.168.56.109]
ok: [192.168.56.110]
ok: [192.168.56.111]

TASK [update repository index] ***************************************************
skipping: [192.168.56.111]
changed: [192.168.56.110]
changed: [192.168.56.109]

TASK [install apache2 package] ***************************************************
skipping: [192.168.56.111]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [add PHP support for apache] ************************************************
skipping: [192.168.56.111]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [update repository index] ***************************************************
skipping: [192.168.56.109]
skipping: [192.168.56.110]
skipping: [192.168.56.111]
```



```
TASK [update repository index] ***************************************************************************************************
skipping: [192.168.56.109]
skipping: [192.168.56.110]
skipping: [192.168.56.111]

TASK [install apache2 package] ***************************************************************************************************
skipping: [192.168.56.109]
skipping: [192.168.56.110]
skipping: [192.168.56.111]

TASK [add PHP support for apache] ************************************************************************************************
skipping: [192.168.56.109]
skipping: [192.168.56.110]
skipping: [192.168.56.111]

PLAY RECAP ***********************************************************************************************************************
192.168.56.109             : ok=4    changed=1    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
192.168.56.110             : ok=4    changed=1    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
192.168.56.111             : ok=1    changed=0    unreachable=0    failed=0    skipped=6    rescued=0    ignored=0
```

It resulted in a lot of skips even though it was stated that the centos vm should update and install.

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.

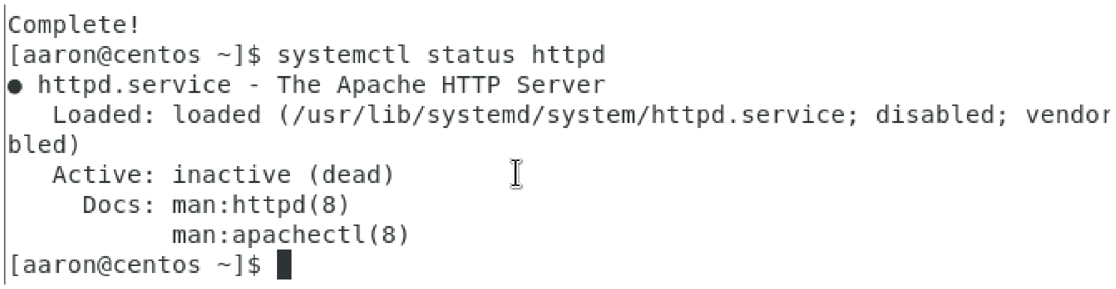


Unable to connect

An error occurred during a connection to 192.168.56.111.

- 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.

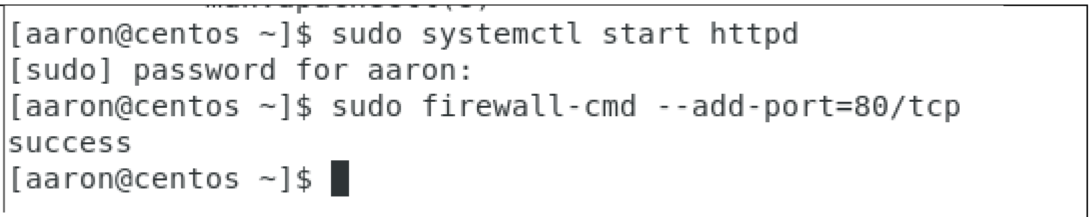
5.1 To activate, go to the CentOS VM terminal and enter the following:
systemctl status httpd
The result of this command tells you that the service is inactive.



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

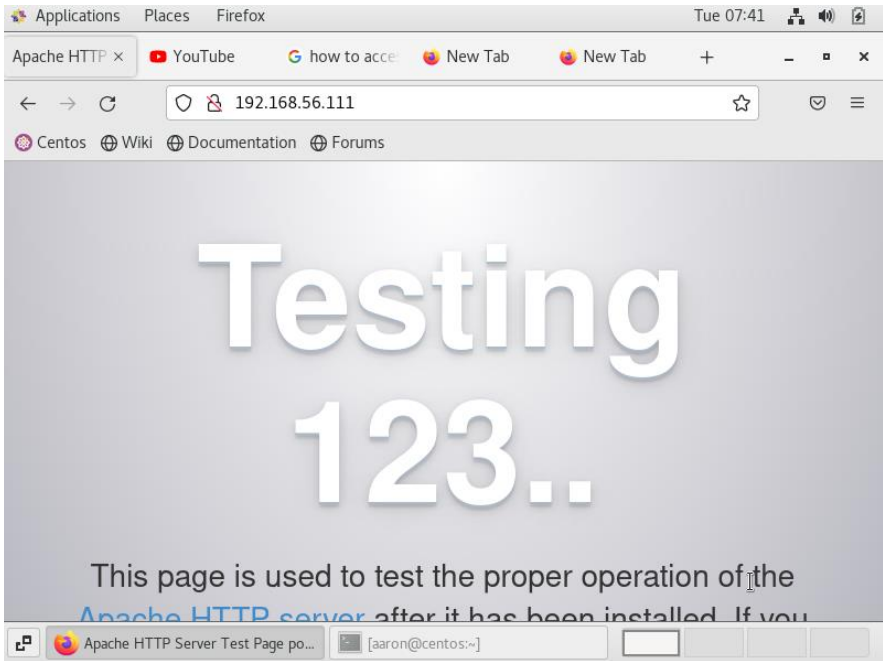
5.2 Issue the following command to start the service:

sudo systemctl start httpd
(When prompted, enter the sudo password)
sudo firewall-cmd --add-port=80/tcp
(The result should be a success)



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

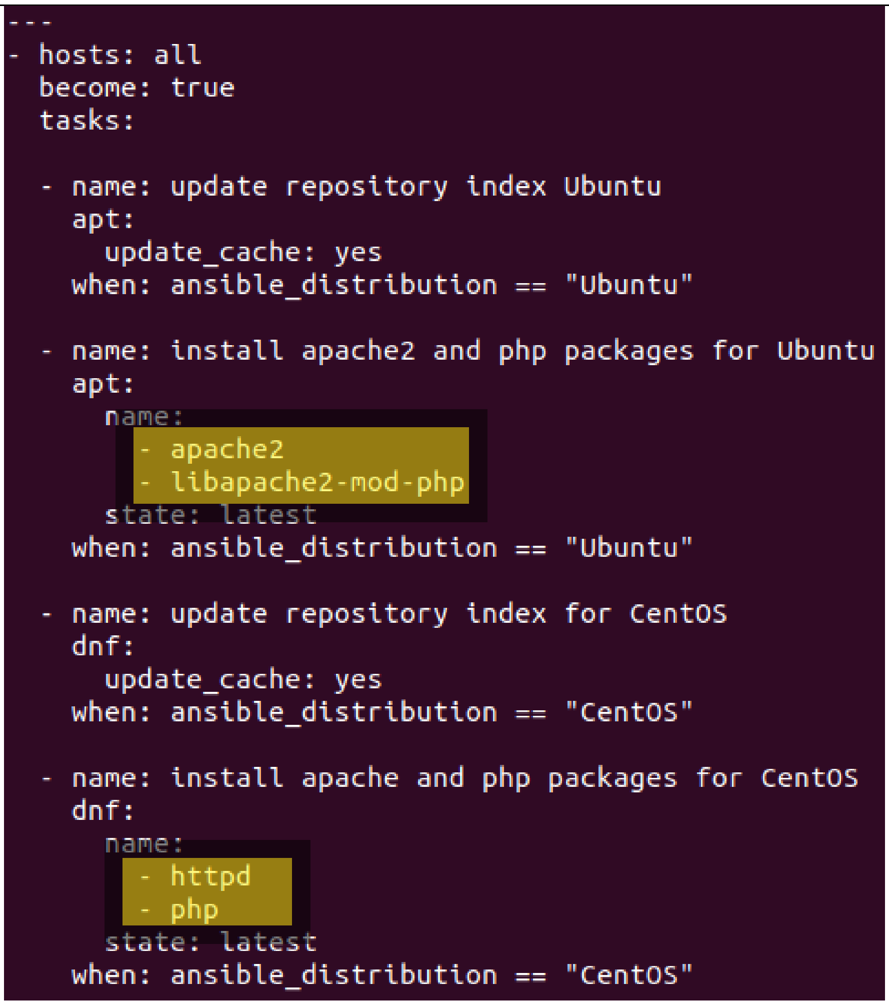


This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you

Task 2: Refactoring playbook

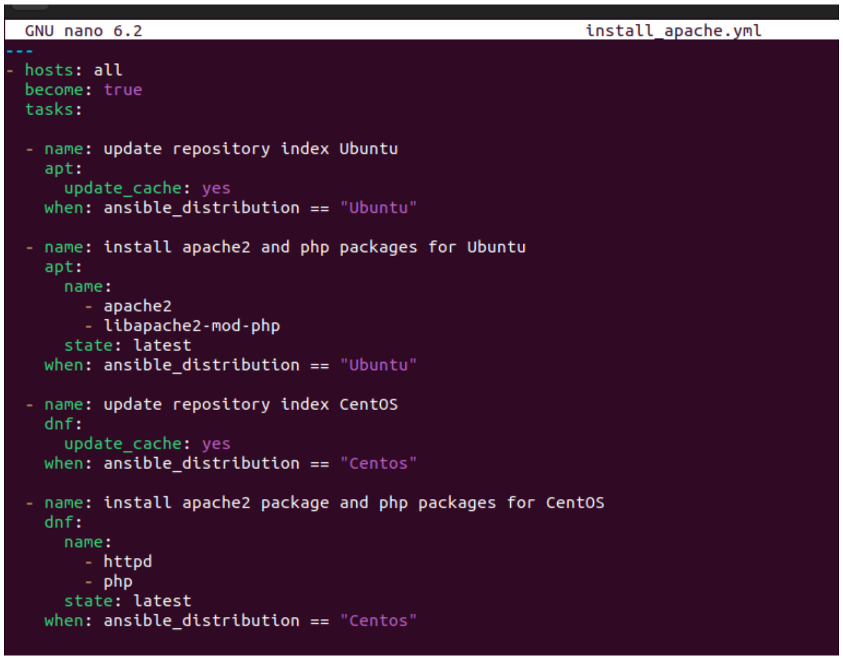
This time, we want to make sure that our playbook is efficient and that the codes are easier to read. This will also makes run ansible more quickly if it has to execute fewer tasks to do the same thing.

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:



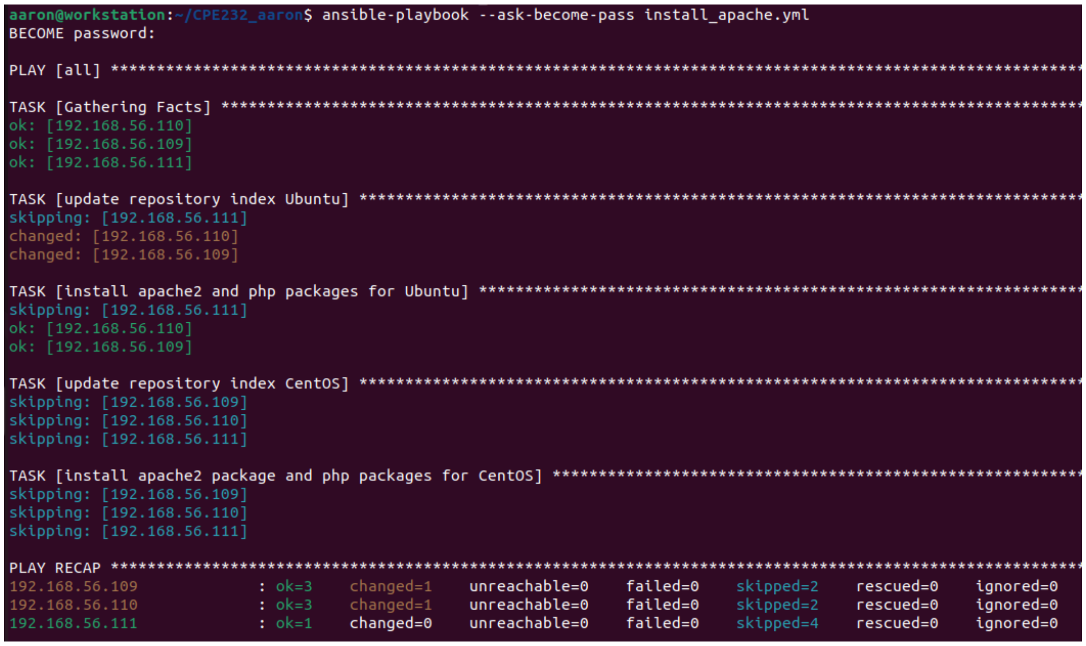
```
---
- hosts: all
  become: true
  tasks:

  - 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.



```
  GNU nano 6.2                                            install_apache.yml
---
- 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 CentOS
    dnf:
      update_cache: yes
    when: ansible_distribution == "Centos"

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

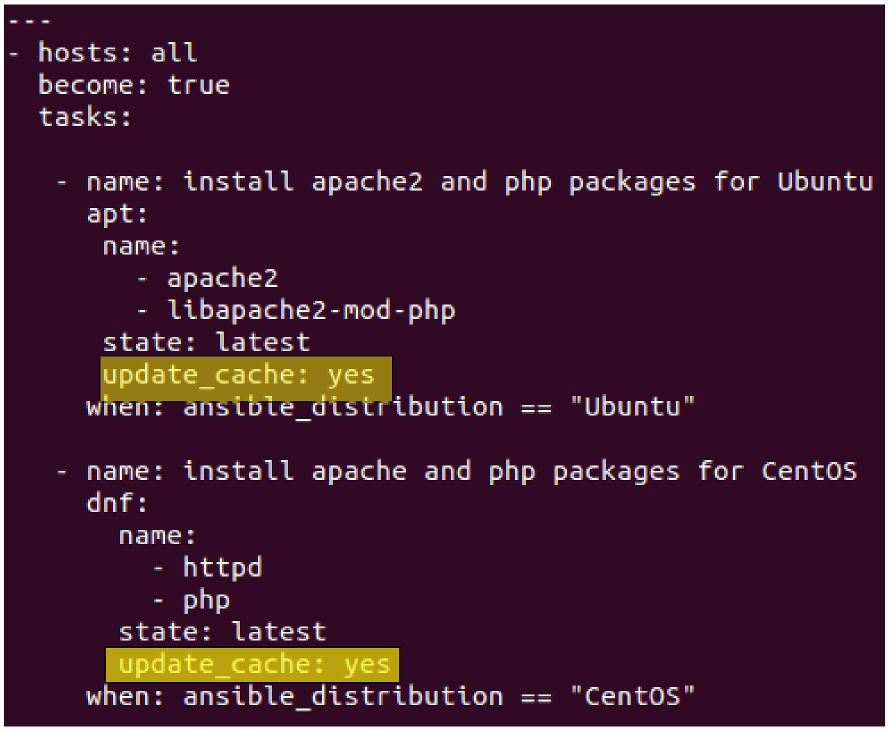
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
aaron@workstation:~/CPE232_aaron$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ****************************************************************************
ok: [192.168.56.110]
ok: [192.168.56.109]
ok: [192.168.56.111]

TASK [update repository index Ubuntu] *************************************************************
skipping: [192.168.56.111]
changed: [192.168.56.110]
changed: [192.168.56.109]

TASK [install apache2 and php packages for Ubuntu] ************************************************
skipping: [192.168.56.111]
ok: [192.168.56.110]
ok: [192.168.56.109]

TASK [update repository index CentOS] *************************************************************
skipping: [192.168.56.109]
skipping: [192.168.56.110]
skipping: [192.168.56.111]

TASK [install apache2 package and php packages for CentOS] ****************************************
skipping: [192.168.56.109]
skipping: [192.168.56.110]
skipping: [192.168.56.111]

PLAY RECAP ****************************************************************************************
192.168.56.109             : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.110             : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.111             : ok=1    changed=0    unreachable=0    failed=0    skipped=4    rescued=0    ignored=0
```

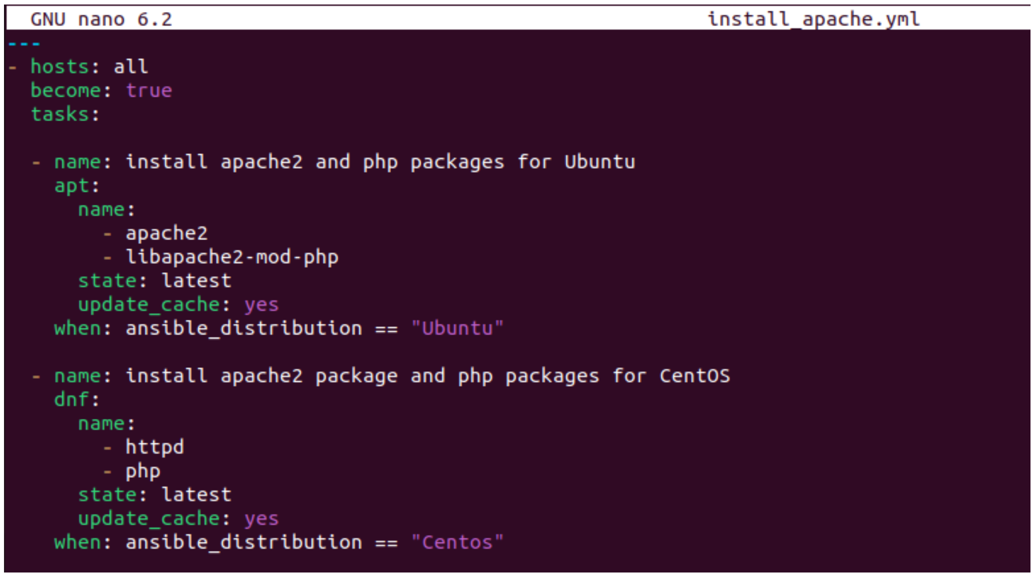
The result is the same the only difference is that the command is shorter.

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:



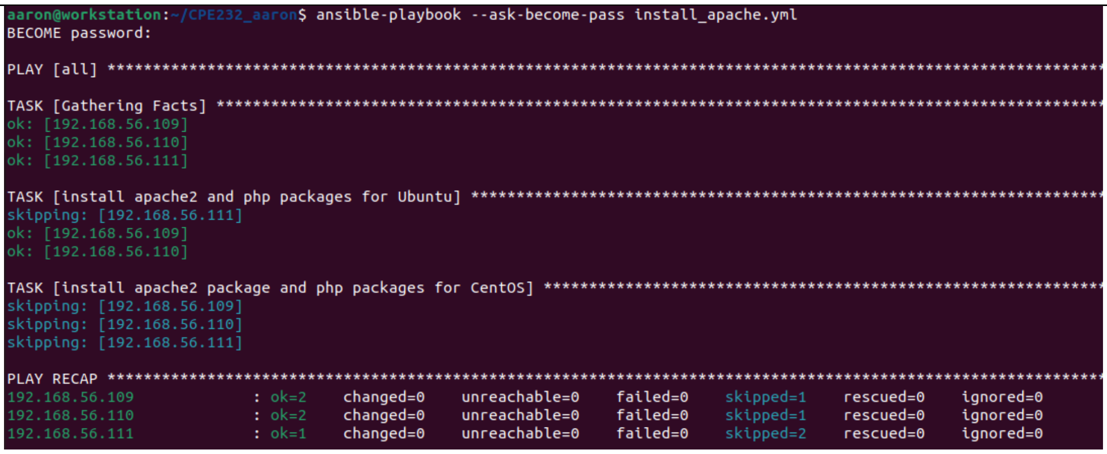
```
---
- hosts: all
  become: true
  tasks:

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

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

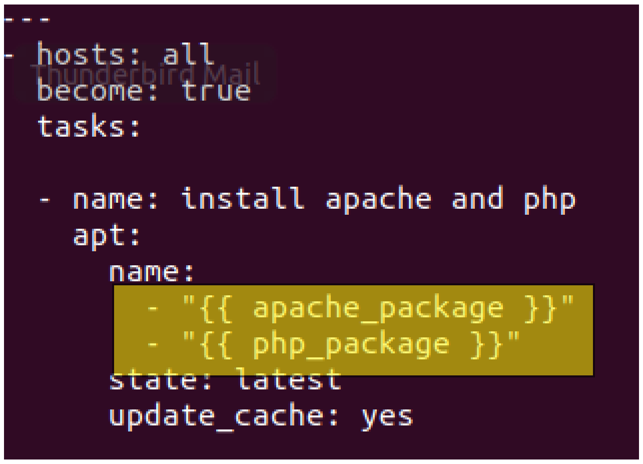
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



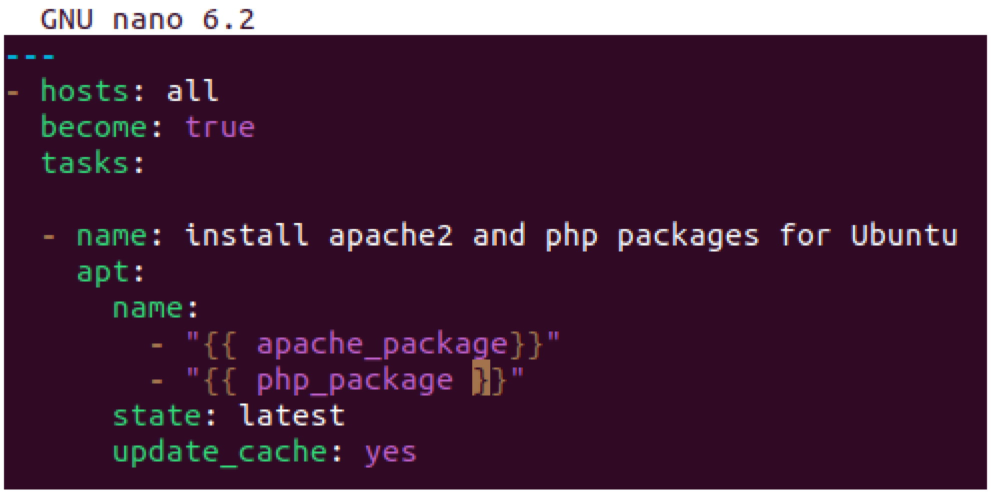
```
aaron@workstation:~/CPE232_aaron$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************************************************
ok: [192.168.56.109]
ok: [192.168.56.110]
ok: [192.168.56.111]

TASK [install apache2 and php packages for Ubuntu] *******************************************************
skipping: [192.168.56.111]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [install apache2 package and php packages for CentOS] ***********************************************
skipping: [192.168.56.109]
skipping: [192.168.56.110]
skipping: [192.168.56.111]

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

The result was the same but the code was now shorter.

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.

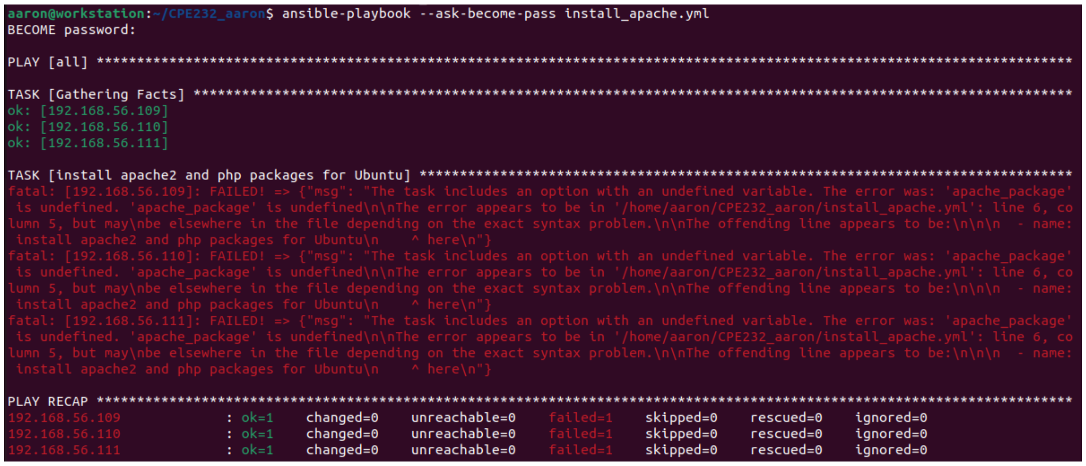


```
---
- hosts: all
  become: true
  tasks:

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



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

  - name: install apache2 and php packages for Ubuntu
    apt:
      name:
        - "{{ apache_package}}"
        - "{{ php_package }}"
      state: latest
      update_cache: yes
```

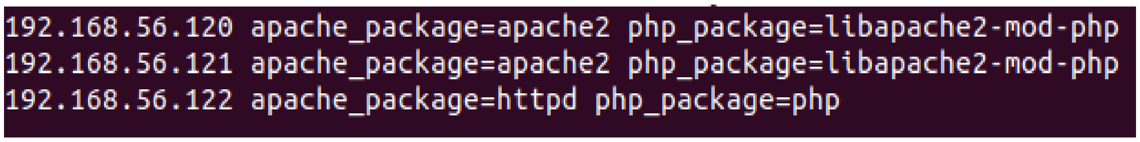
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
aaron@workstation:~/CPE232_aaron$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************************************************************
ok: [192.168.56.109]
ok: [192.168.56.110]
ok: [192.168.56.111]

TASK [install apache2 and php packages for Ubuntu] *******************************************************************
fatal: [192.168.56.109]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'apache_package'
 is undefined. 'apache_package' is undefined\n\nThe error appears to be in '/home/aaron/CPE232_aaron/install_apache.yml': line 6, co
lumn 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 apache2 and php packages for Ubuntu\n    ^ here\n"}
fatal: [192.168.56.110]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'apache_package'
 is undefined. 'apache_package' is undefined\n\nThe error appears to be in '/home/aaron/CPE232_aaron/install_apache.yml': line 6, co
lumn 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 apache2 and php packages for Ubuntu\n    ^ here\n"}
fatal: [192.168.56.111]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'apache_package'
 is undefined. 'apache_package' is undefined\n\nThe error appears to be in '/home/aaron/CPE232_aaron/install_apache.yml': line 6, co
lumn 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 apache2 and php packages for Ubuntu\n    ^ here\n"}

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

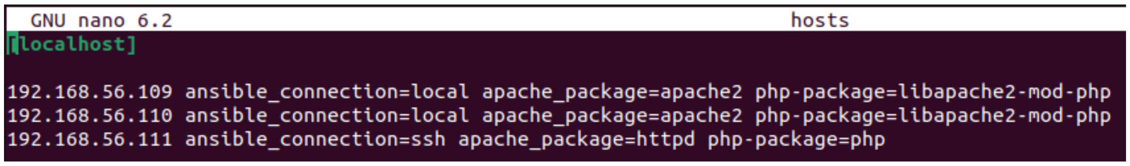
The result was a failure but the connection is still established.

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:

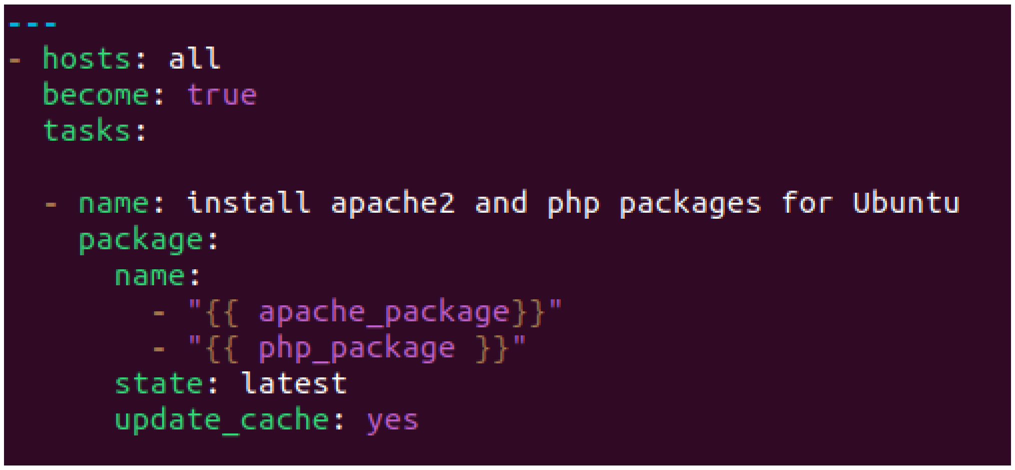


```
192.168.56.120 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.121 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.122 apache_package=httpd php_package=php
```

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



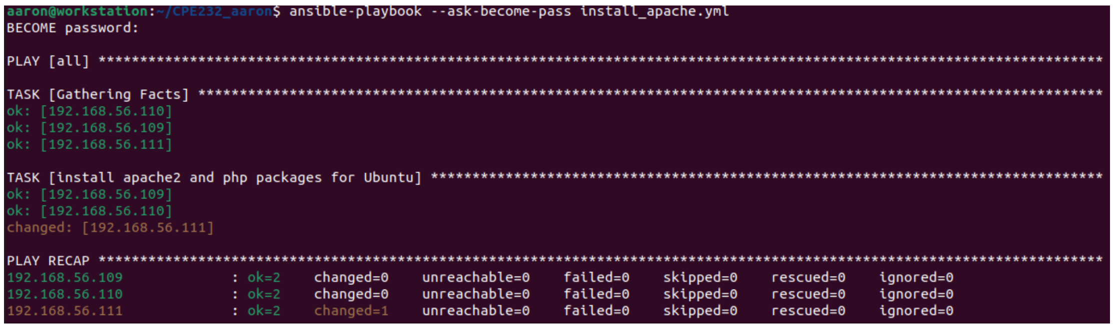
```
  GNU nano 6.2                                                hosts
[localhost]

192.168.56.109 ansible_connection=local apache_package=apache2 php-package=libapache2-mod-php
192.168.56.110 ansible_connection=local apache_package=apache2 php-package=libapache2-mod-php
192.168.56.111 ansible_connection=ssh apache_package=httpd php-package=php
```



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

  - name: install apache2 and php packages for Ubuntu
    package:
      name:
        - "{{ apache_package}}"
        - "{{ php_package }}"
      state: latest
      update_cache: yes
```

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.



```
aaron@workstation:~/CPE232_aaron$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************************************************************
ok: [192.168.56.110]
ok: [192.168.56.109]
ok: [192.168.56.111]

TASK [install apache2 and php packages for Ubuntu] *****************************************************************************
ok: [192.168.56.109]
ok: [192.168.56.110]
changed: [192.168.56.111]

PLAY RECAP *********************************************************************************************************************
192.168.56.109             : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.110             : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.111             : ok=2    changed=1    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 of playbook codes is important for several reasons. Firstly, it enhances code readability and maintainability, making it easier for both the original developer and others to understand and modify the code as needed. This improves collaboration within a team and reduces the likelihood of errors. Secondly, refactoring can lead to more efficient code by identifying and eliminating redundant or inefficient operations, resulting in improved performance. Additionally, refactoring ensures that the playbook adheres to best practices and coding standards, enhancing its reliability and reducing the risk of vulnerabilities. Overall, playbook code refactoring is a crucial step in the software development process that promotes code quality, reliability, and efficiency.
2. **When do we use the “when” command in playbook?**
   The "when" command in a playbook is used to conditionally control the execution of specific tasks or roles based on certain conditions or variables. It allows playbook authors to define under what circumstances a particular task should be executed, making the playbook more flexible and responsive to varying conditions. For example, "when" can be employed to execute tasks only if certain files or packages are present, specific facts are true, or certain variables meet predefined criteria. This conditional logic enables playbooks to adapt to different environments and ensures

that tasks are executed appropriately, reducing unnecessary work and streamlining automation processes.