Name: Renier L. Lope	Date Performed: 09/24/2024
Course/Section: CPE212 – CPE31S2	Date Submitted: 09/24/2024
Instructor: Engr. Robin Valenzuela	Semester & SY: 1ˢᵗSem(2024-2025)

<div align="center">

Activity 5: Consolidating Playbook plays

</div>

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 (*CPE212_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?
   – **It outputs 'Already up to date.', because git pull command will update the local repository to match the remote repository.**

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.
-   **It selects what type of server it would distribute the task, thereby skipping CentOS only executing on Ubuntu servers.**

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.

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)

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

Task 1 Outputs:

Figure 1.1: Issuing command git pull.

Figure 1.2: Adding CentOS to inventory.

Figure 1.3: Execute playbook install_apache.yml with CentOS.

Figure 1.4: Editing install_apache.yml

```
rnrlope@workstation:~/CPE212_LOPE$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ********************************************************
ok: [server2]
ok: [centOS]
ok: [server1]

TASK [update repository index] ***********************************************
skipping: [centOS]
changed: [server2]
changed: [server1]

TASK [install apache2 package] ***********************************************
skipping: [centOS]
ok: [server2]
ok: [server1]

TASK [add PHP support for apache] ********************************************
skipping: [centOS]
ok: [server1]
ok: [server2]

PLAY RECAP ********************************************************************
centOS                     : ok=1    changed=0    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
server1                    : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
server2                    : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

rnrlope@workstation:~/CPE212_LOPE$
```

Figure 1.5: Execute playbook install_apache.yml after doing (figure 1.4).

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

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

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

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

Figure 1.6: Editing playbook install_apache.yml to distribute task to CentOS.

```
PLAY RECAP ********************************************************************
centOS                     : ok=4    changed=2    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
server1                    : ok=4    changed=1    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
server2                    : ok=4    changed=1    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0

rnrlope@workstation:~/CPE212_LOPE$
```
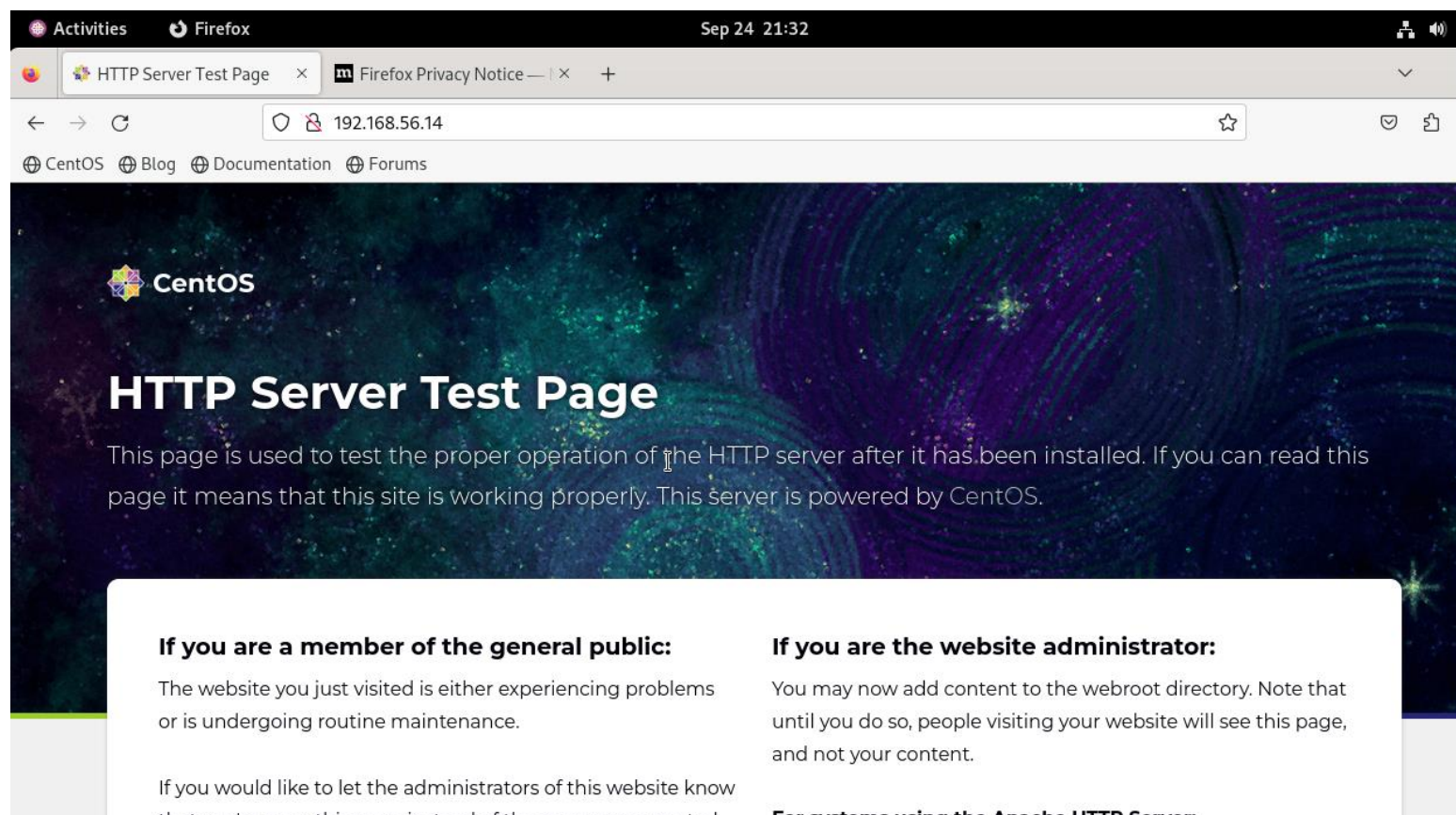
Figure 1.7: Output of the edited playbook install_apache.yml refer to (figure 1.6).

Figure 1.8: Activating httpd service or Apache HTTP server in CentOS.

Figure 1.9: Verifying if the Apache HTTP service is running.

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:

```yaml
---
- 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.
- **It runs perfectly, the same as before consolidation.**

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.
- **It runs perfectly, the same as before consolidation.**

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.
- **It runs nothing because the variable we place points to nothing.**

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.

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.
- **It runs perfectly, without the skipping.**

Task 2 Outputs:

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

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

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

Figure 2.1: Consolidating two task that install packages into 1 task for both Ubuntu and CentOS.

```
PLAY RECAP *********************************************************************
centOS                     : ok=3    changed=0    unreachable=0    failed=0    skipped=2    rescued=0
 ignored=0
server1                    : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0
 ignored=0
server2                    : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0
 ignored=0

rnrlope@workstation:~/CPE212_LOPE$
```

Figure 2.2: Output after consolidation of two task into 1 task refer to (figure 2.1).

```yaml
---
- 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 and php packages in CentOS
      dnf:
        name:
          - httpd
          - php
        state: latest
        update_cache: yes
      when: ansible_distribution == "CentOS"
```

Figure 2.3: Consolidation of every task in just 2 task.

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

server1                    : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0

server2                    : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0


rnrlope@workstation:~/CPE212_LOPE$
```

Figure 2.4: Output after consolidation of every task in just 1 task refer to (figure 2.3).

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

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

Figure 2.5: Consolidation of every task in just 1 task.

```
PLAY RECAP *****************************************************************************
centOS                    : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0

server1                   : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0

server2                   : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0


rnrlope@workstation:~/CPE212_LOPE$
```

Figure 2.6: Output of figure 2.5.

```
[ManageNodes]
server1 apache_package=apache2 php_package=libapache2-mod-php
server2 apache_package=apache2 php_package=libapache2-mod-php
centOS apache_package=httpd php_package=php
```

Figure 2.7: Giving values to variables in inventory next to IP address/hostnames of servers.

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

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

Figure 2.8: Editing install_apache.yml. Changing apt to package to use whatever package manager the underlying host or the target server uses.

```
rnrlope@workstation:~/CPE212_LOPE$ nano install_apache.yml
rnrlope@workstation:~/CPE212_LOPE$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ********************************************************************
ok: [server2]
ok: [server1]
ok: [centOS]

TASK [install apache2 and php packages] **************************************************
ok: [centOS]
ok: [server2]
ok: [server1]

PLAY RECAP ********************************************************************************
centOS                    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

server1                   : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

server2                   : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0


rnrlope@workstation:~/CPE212_LOPE$
```

Figure 2.9: Output with all the changes in inventory and install_apache.yml
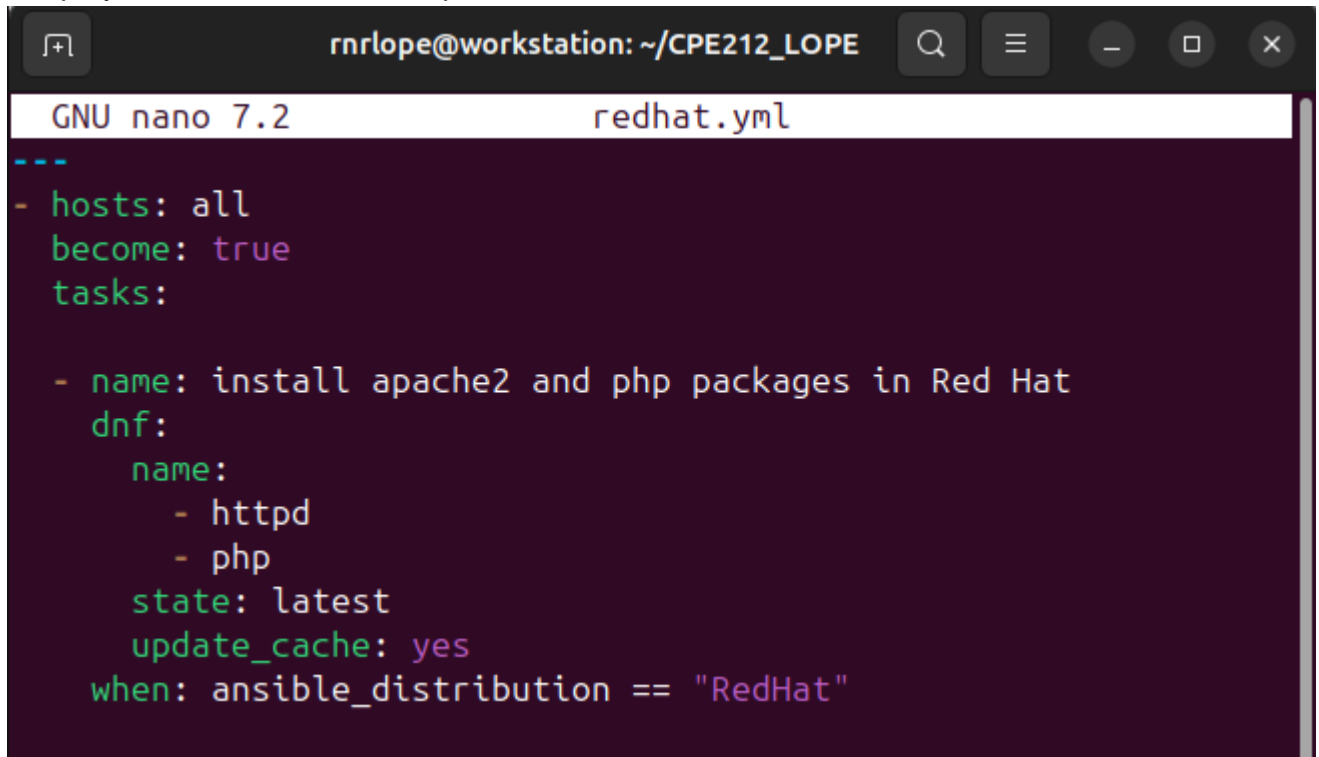
GIT PUSH:

```
rnrlope@workstation:~/CPE212_LOPE$ git add inventory
rnrlope@workstation:~/CPE212_LOPE$ git add install_apache.yml
rnrlope@workstation:~/CPE212_LOPE$ git commit -m "Activity5"
[main dc04ff0] Activity5
 2 files changed, 10 insertions(+), 12 deletions(-)
rnrlope@workstation:~/CPE212_LOPE$ git push origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 2 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 530 bytes | 530.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:RenierCode/CPE212_LOPE.git
   6181ca0..dc04ff0  main -> main
rnrlope@workstation:~/CPE212_LOPE$
```

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?
   - **To reduce code complexity, or hard-to-understand code, resulting in cleaner and easy to understand codes.**
2. When do we use the "when" command in playbook?
   - **If we want a specific task to only run in a specific server type. For example: when: ansible_distribution == "RedHat", task with these arguments will only run in Red Hat servers.**