Name: Bernice M. Peña	**Date Performed:** 09/08/2023
Course/Section: Managing Enterprise Servers / CPE31S5	**Date Submitted:** 09/12/2023
Instructor: Engr. Roman Richard	**Semester and SY:** 1st semester, S.Y. 2023-2024

Activity 4: Running Elevated Ad hoc Commands

1. Objectives:

1.1 Use commands that makes changes to remote machines

1.2 Use playbook in automating ansible commands

2. Discussion:

Provide screenshots for each task.

Elevated Ad hoc commands

So far, we have not performed ansible commands that makes changes to the remote servers. We manage to gather facts and connect to the remote machines, but we still did not make changes on those machines. In this activity, we will learn to use commands that would install, update, and upgrade packages in the remote machines. We will also create a playbook that will be used for automations.

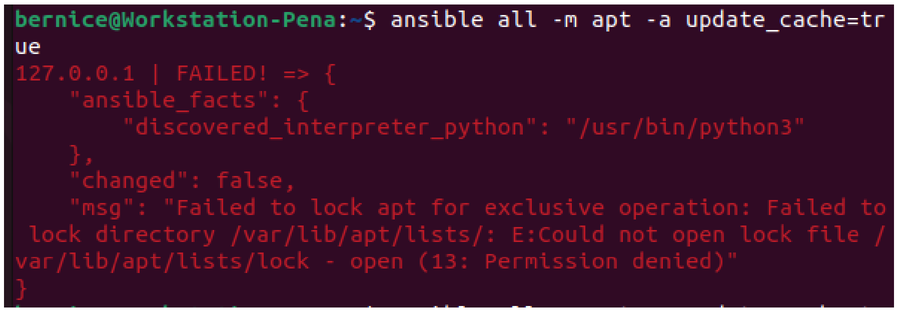
Playbooks record and execute **Ansible**'s configuration, deployment, and orchestration functions. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. If Ansible modules are the tools in your workshop, playbooks are your instruction manuals, and your inventory of hosts are your raw material. At a basic level, playbooks can be used to manage configurations of and deployments to remote machines. At a more advanced level, they can sequence multi-tier rollouts involving rolling updates, and can delegate actions to other hosts, interacting with monitoring servers and load balancers along the way. You can check this documentation if you want to learn more about playbooks. Working with playbooks — Ansible Documentation

Task 1: Run elevated ad hoc commands

1. Locally, we use the command *sudo apt update* when we want to download package information from all configured resources. The sources often defined in /etc/apt/sources.list file and other files located in /etc/apt/sources.list.d/ directory. So, when you run update command, it downloads the package information from the Internet. It is useful to get info on an updated version of packages or their dependencies. We can only run an apt update command in a remote machine. Issue the following command:

ansible all -m apt -a update_cache=true

What is the result of the command? Is it successful?

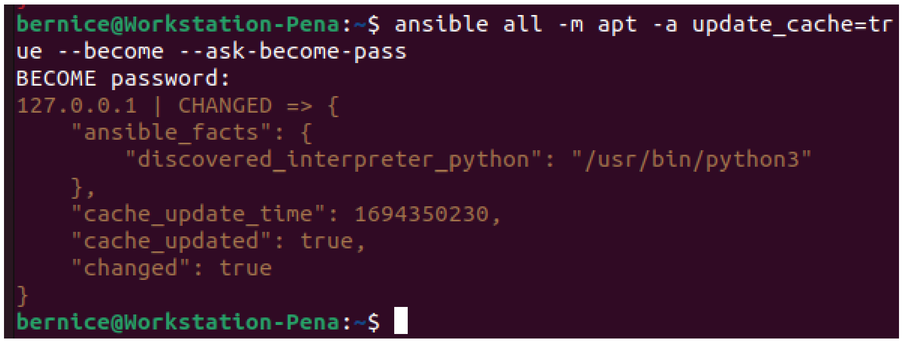


```
bernice@Workstation-Pena:~$ ansible all -m apt -a update_cache=tr
ue
127.0.0.1 | FAILED! => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "msg": "Failed to lock apt for exclusive operation: Failed to
 lock directory /var/lib/apt/lists/: E:Could not open lock file /
var/lib/apt/lists/lock - open (13: Permission denied)"
}
```

After executing the command, it resulted in "FAILED"

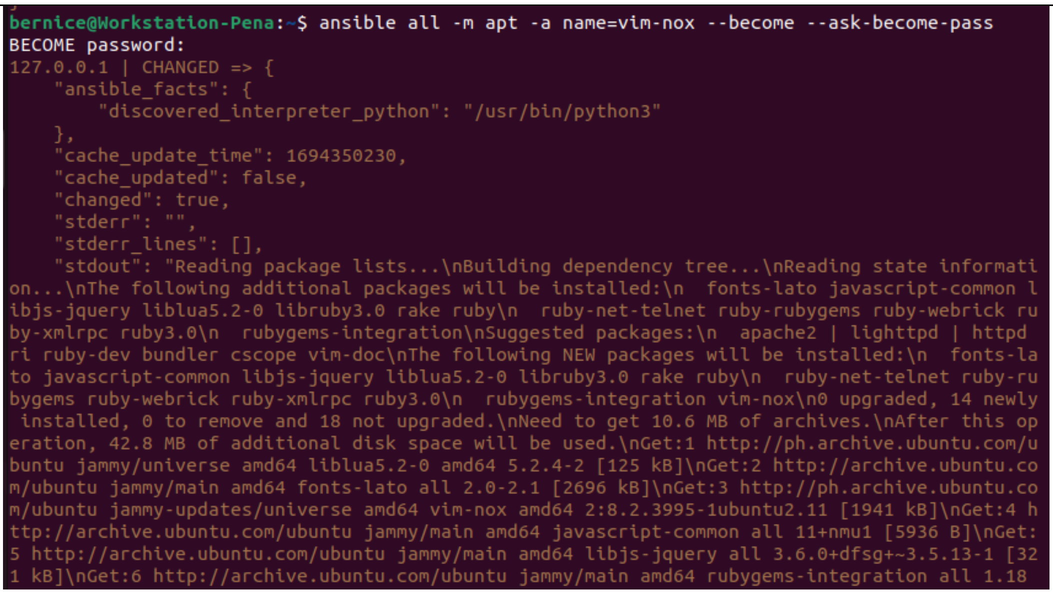
Try editing the command and add something that would elevate the privilege. Issue the command *ansible all -m apt -a update_cache=true --become --ask-become-pass.* Enter the sudo password when prompted. You will notice now that the output of this command is a success. The *update_cache=true* is the same thing as running *sudo apt update*. The --become command elevate the privileges and the *--ask-become-pass* asks for the password. For now, even if we only have changed the packaged index, we were able to change something on the remote server.

You may notice after the second command was executed, the status is CHANGED compared to the first command, which is FAILED.



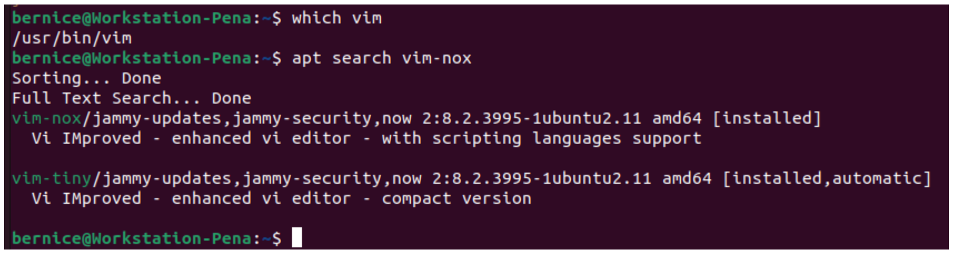
```
bernice@Workstation-Pena:~$ ansible all -m apt -a update_cache=tr
ue --become --ask-become-pass
BECOME password:
127.0.0.1 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1694350230,
    "cache_updated": true,
    "changed": true
}
bernice@Workstation-Pena:~$
```

2. Let's try to install VIM, which is an almost compatible version of the UNIX editor Vi. To do this, we will just changed the module part in 1.1 instruction. Here is the command: *ansible all -m apt -a name=vim-nox --become --ask-become-pass.* The command would take some time after typing the password because the local machine instructed the remote servers to actually install the package.



```
bernice@Workstation-Pena:~$ ansible all -m apt -a name=vim-nox --become --ask-become-pass
BECOME password:
127.0.0.1 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1694350230,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading state informati
on...\nThe following additional packages will be installed:\n  fonts-lato javascript-common l
ibjs-jquery liblua5.2-0 libruby3.0 rake ruby\n  ruby-net-telnet ruby-rubygems ruby-webrick ru
by-xmlrpc ruby3.0\n  rubygems-integration\nSuggested packages:\n  apache2 | lighttpd | httpd
ri ruby-dev bundler cscope vim-doc\nThe following NEW packages will be installed:\n  fonts-la
to javascript-common libjs-jquery liblua5.2-0 libruby3.0 rake ruby\n  ruby-net-telnet ruby-ru
bygems ruby-webrick ruby-xmlrpc ruby3.0\n  rubygems-integration vim-nox\n0 upgraded, 14 newly
 installed, 0 to remove and 18 not upgraded.\nNeed to get 10.6 MB of archives.\nAfter this op
eration, 42.8 MB of additional disk space will be used.\nGet:1 http://ph.archive.ubuntu.com/u
buntu jammy/universe amd64 liblua5.2-0 amd64 5.2.4-2 [125 kB]\nGet:2 http://archive.ubuntu.co
m/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1 [2696 kB]\nGet:3 http://ph.archive.ubuntu.co
m/ubuntu jammy-updates/universe amd64 vim-nox amd64 2:8.2.3995-1ubuntu2.11 [1941 kB]\nGet:4 h
ttp://archive.ubuntu.com/ubuntu jammy/main amd64 javascript-common all 11+nmu1 [5936 B]\nGet:
5 http://archive.ubuntu.com/ubuntu jammy/main amd64 libjs-jquery all 3.6.0+dfsg+~3.5.13-1 [32
1 kB]\nGet:6 http://archive.ubuntu.com/ubuntu jammy/main amd64 rubygems-integration all 1.18
```

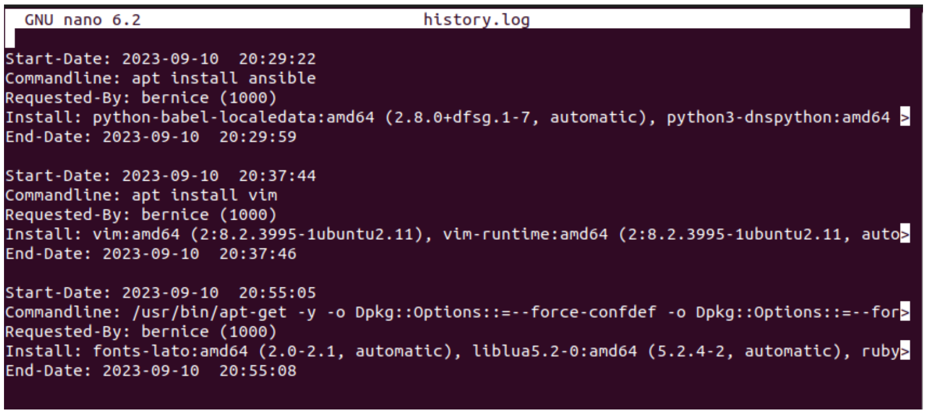
2.1 Verify that you have installed the package in the remote servers. Issue the command *which vim* and the command *apt search vim-nox* respectively. Was the command successful?



```
bernice@Workstation-Pena:~$ which vim
/usr/bin/vim
bernice@Workstation-Pena:~$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/jammy-updates,jammy-security,now 2:8.2.3995-1ubuntu2.11 amd64 [installed]
  Vi IMproved - enhanced vi editor - with scripting languages support

vim-tiny/jammy-updates,jammy-security,now 2:8.2.3995-1ubuntu2.11 amd64 [installed,automatic]
  Vi IMproved - enhanced vi editor - compact version

bernice@Workstation-Pena:~$
```

The command was successful since the vim was found at /usr/bin/vim and the second command confirmed that it is installed.

2.2 Check the logs in the servers using the following commands: *cd /var/log*. After this, issue the command *ls,* go to the folder *apt* and open history.log. Describe what you see in the history.log.

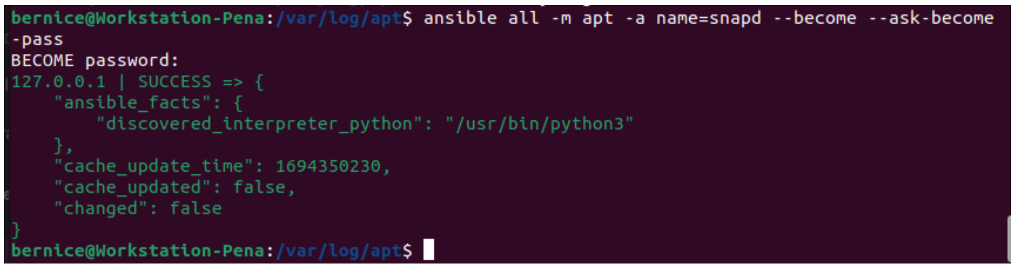


```
  GNU nano 6.2                              history.log

Start-Date: 2023-09-10  20:29:22
Commandline: apt install ansible
Requested-By: bernice (1000)
Install: python-babel-localedata:amd64 (2.8.0+dfsg.1-7, automatic), python3-dnspython:amd64 >
End-Date: 2023-09-10  20:29:59

Start-Date: 2023-09-10  20:37:44
Commandline: apt install vim
Requested-By: bernice (1000)
Install: vim:amd64 (2:8.2.3995-1ubuntu2.11), vim-runtime:amd64 (2:8.2.3995-1ubuntu2.11, auto>
End-Date: 2023-09-10  20:37:46

Start-Date: 2023-09-10  20:55:05
Commandline: /usr/bin/apt-get -y -o Dpkg::Options::=--force-confdef -o Dpkg::Options::=--for>
Requested-By: bernice (1000)
Install: fonts-lato:amd64 (2.0-2.1, automatic), liblua5.2-0:amd64 (5.2.4-2, automatic), ruby>
End-Date: 2023-09-10  20:55:08
```

In the history.log file, there are different records of package installation and upgrade activities performed on the system. These activities were initiated by the user bernice (me). On September 10, 2023, at 20:29:22, the ansible package was installed, along with its dependencies. Later at 20:37:44, the vim package and its runtime components were installed, this followed by a series of packages including fonts-lato, liblua5.2-0, and ruby, among others were installed. Each log entry includes a timestamp, my activities, and the specific packages installed.

3. This time, we will install a package called snapd. Snap is pre-installed in Ubuntu system. However, our goal is to create a command that checks for the latest installation package.

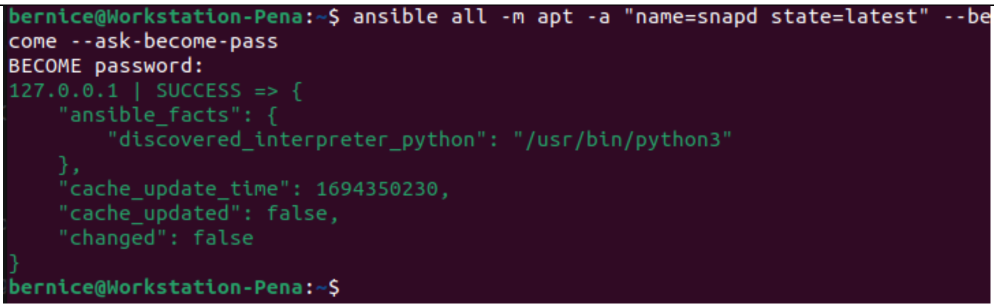
   3.1 Issue the command: *ansible all -m apt -a name=snapd --become --ask-become-pass*
   Can you describe the result of this command? Is it a success? Did it change anything in the remote servers?



```
bernice@Workstation-Pena:/var/log/apt$ ansible all -m apt -a name=snapd --become --ask-become
-pass
BECOME password:
127.0.0.1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1694350230,
    "cache_updated": false,
    "changed": false
}
bernice@Workstation-Pena:/var/log/apt$ 
```

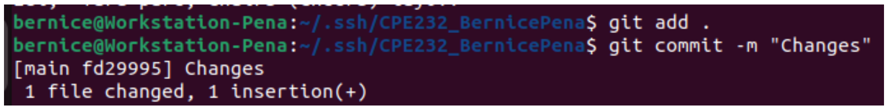
   It was executed successfully and did not target any remote servers as indicated by the fact that changed is false. This means that no changes were made to any remote servers, the command was intended to install or ensure the presence of the snapd package, but it seems that this package was already installed or available and no changes were needed.

   3.2 Now, try to issue this command: *ansible all -m apt -a "name=snapd state=latest" --become --ask-become-pass*
   Describe the output of this command. Notice how we added the command *state=latest* and placed them in double quotations.



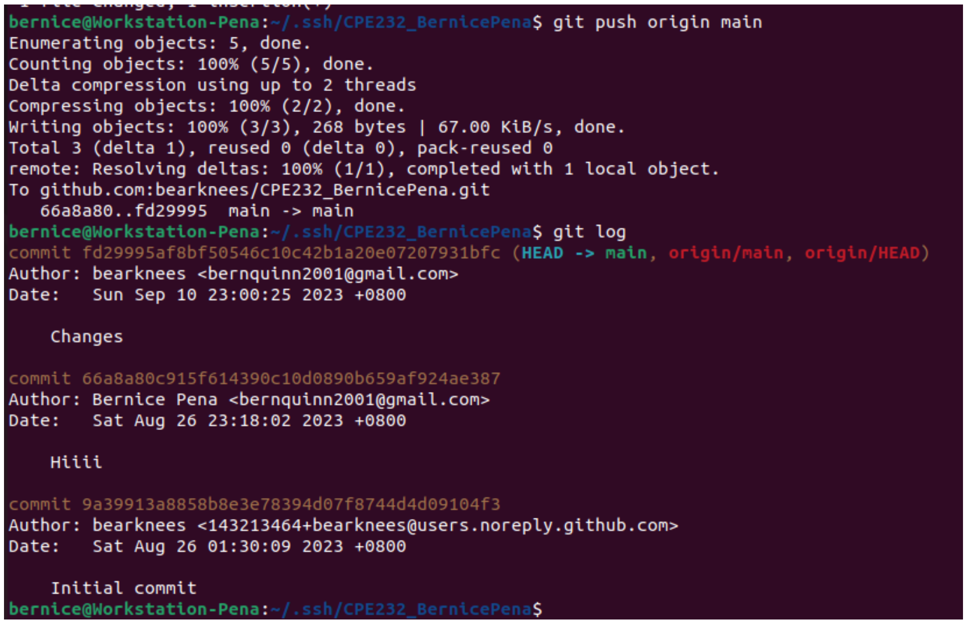
```
bernice@Workstation-Pena:~$ ansible all -m apt -a "name=snapd state=latest" --be
come --ask-become-pass
BECOME password:
127.0.0.1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1694350230,
    "cache_updated": false,
    "changed": false
}
bernice@Workstation-Pena:~$
```

4. At this point, make sure to commit all changes to GitHub.



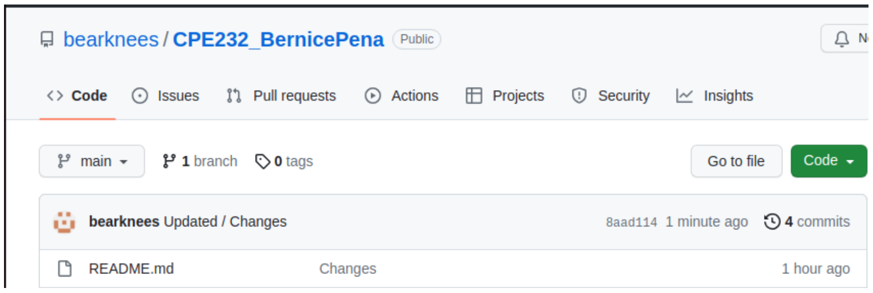
```
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ git add .
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ git commit -m "Changes"
[main fd29995] Changes
 1 file changed, 1 insertion(+)
```

Committing all the changes to GitHub



```
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 268 bytes | 67.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:bearknees/CPE232_BernicePena.git
   66a8a80..fd29995  main -> main
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ git log
commit fd29995af8bf50546c10c42b1a20e07207931bfc (HEAD -> main, origin/main, origin/HEAD)
Author: bearknees <bernquinn2001@gmail.com>
Date:   Sun Sep 10 23:00:25 2023 +0800

    Changes

commit 66a8a80c915f614390c10d0890b659af924ae387
Author: Bernice Pena <bernquinn2001@gmail.com>
Date:   Sat Aug 26 23:18:02 2023 +0800

    Hiiii

commit 9a39913a8858b8e3e78394d07f8744d4d09104f3
Author: bearknees <143213464+bearknees@users.noreply.github.com>
Date:   Sat Aug 26 01:30:09 2023 +0800

    Initial commit
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$
```

Pushing all the changes and confirming the status if they are successfully committed.



bearknees / CPE232_BernicePena Public

Code Issues Pull requests Actions Projects Security Insights

main 1 branch 0 tags Go to file Code

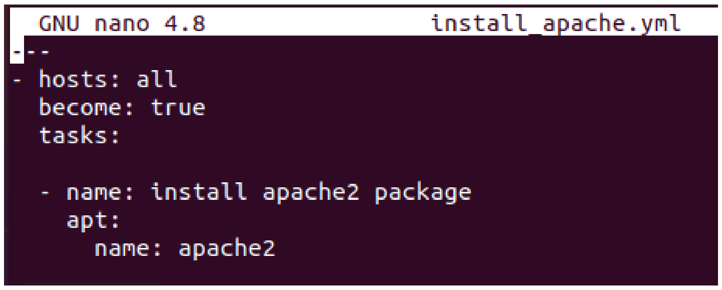
bearknees Updated / Changes 8aad114 1 minute ago 4 commits

README.md Changes 1 hour ago

Task 2: Writing our First Playbook

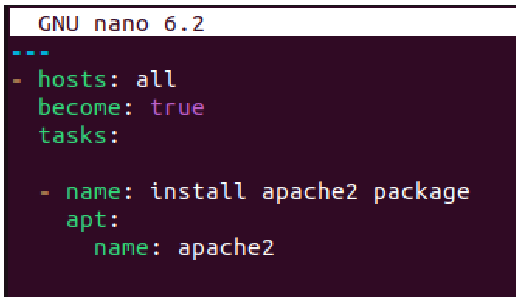
1. With ad hoc commands, we can simplify the administration of remote servers. For example, we can install updates, packages, and applications, etc. However, the real strength of ansible comes from its playbooks. When we write a playbook, we can define the state that we want our servers to be in and the place or commands that ansible will carry out to bring to that state. You can use an editor to create a playbook. Before we proceed, make sure that you are in the directory of the repository that we use in the previous activities (*CPE232_yourname*). Issue the command *nano install_apache.yml*. This will create a playbook file called *install_apache.yml*. The .yml is the basic standard extension for playbook files.

   When the editor appears, type the following:

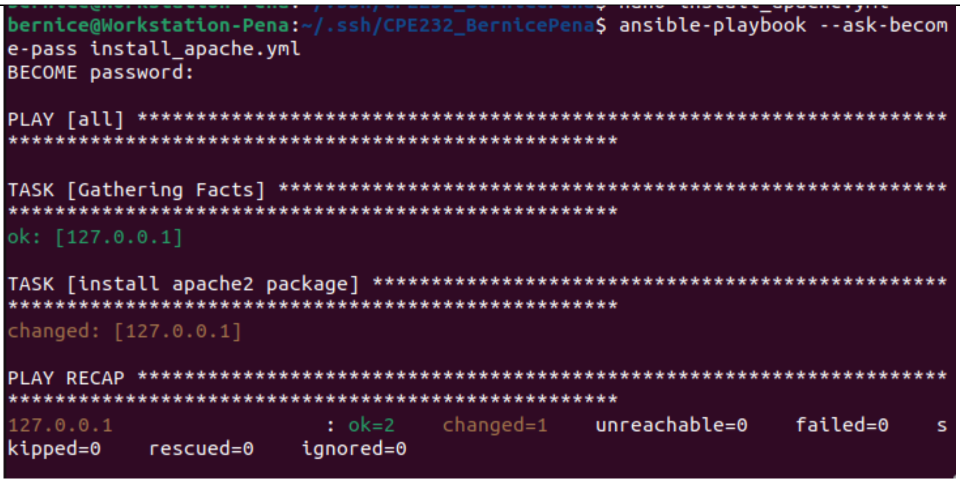
```
  GNU nano 4.8                         install_apache.yml
---
- hosts: all
  become: true
  tasks:

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

   Make sure to save the file. Take note also of the alignments of the texts.

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

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

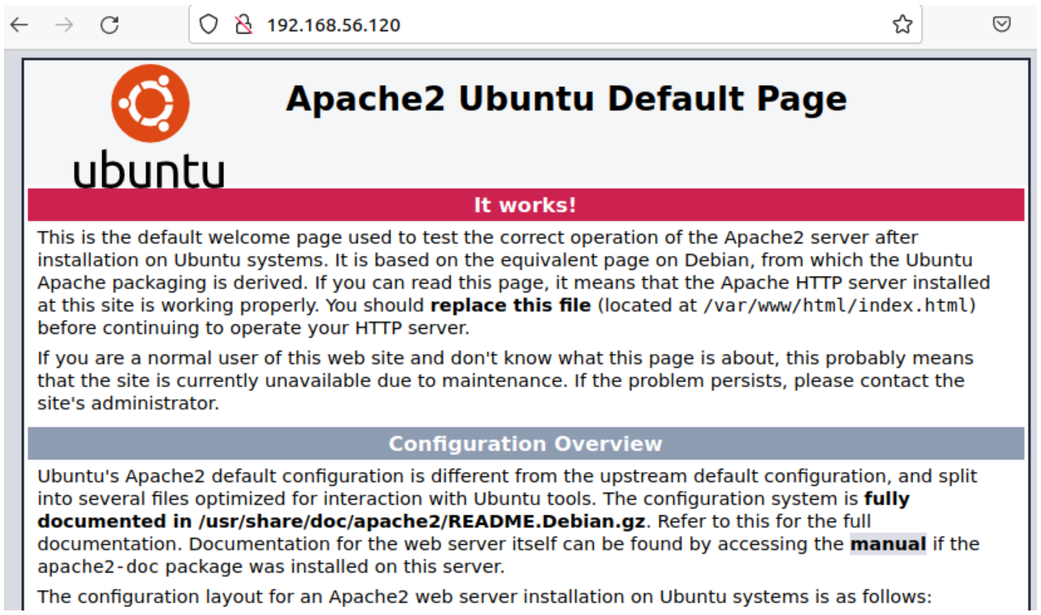
2. Run the yml file using the command: *ansible-playbook --ask-become-pass install_apache.yml.* Describe the result of this command.



```
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ ansible-playbook --ask-becom
e-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] **********************************************************
*****************************************************
ok: [127.0.0.1]

TASK [install apache2 package] **************************************************
*****************************************************
changed: [127.0.0.1]

PLAY RECAP **********************************************************************
*****************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
```

The first task labeled "Gathering Facts," collects essential host information and reports a successful outcome. Regarding with the subsequent task named "install apache2 package," it seems that it carries out the installation of the 'apache2' package using the 'apt'.

3. To verify that apache2 was installed automatically in the remote servers, go to the web browsers on each server and type its IP address. You should see something like this.



192.168.56.120

Apache2 Ubuntu Default Page

ubuntu

It works!

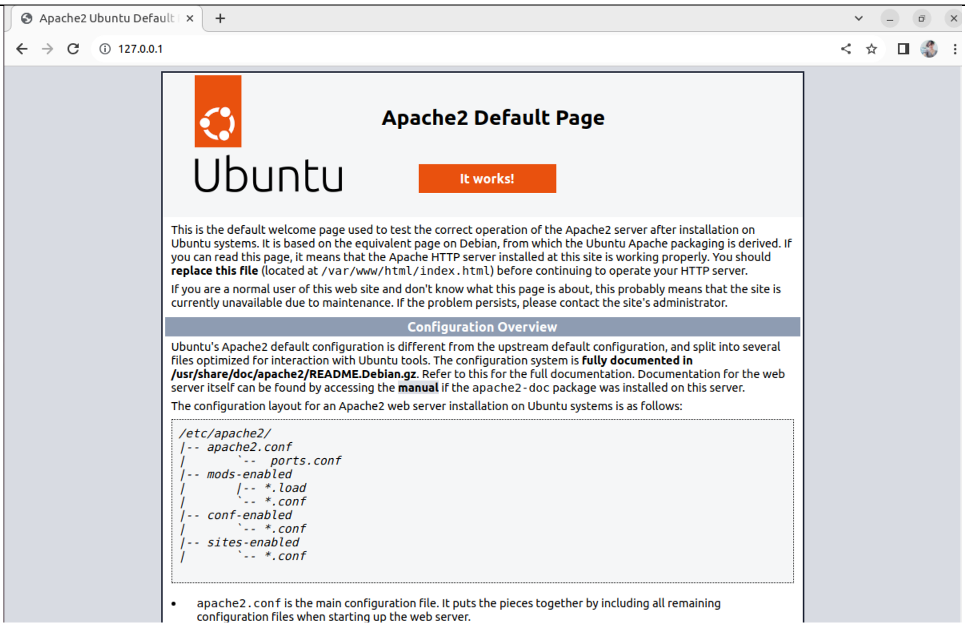
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:



Apache2 Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

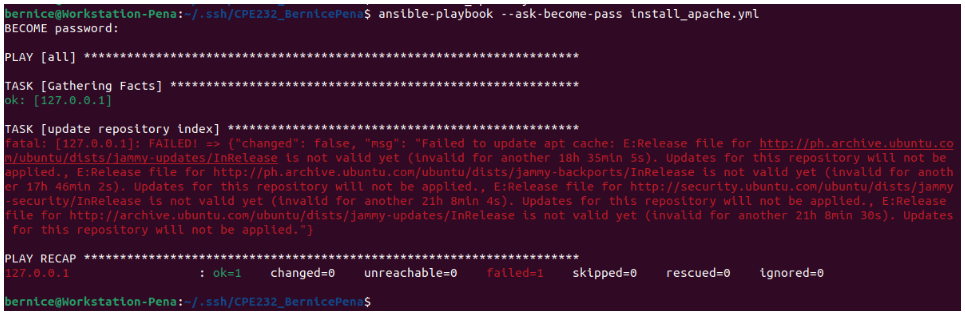
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.

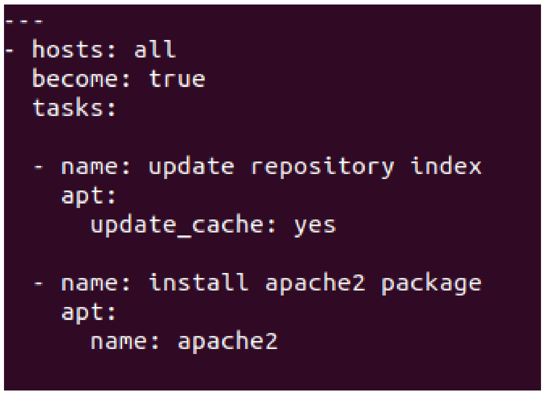
4. Try to edit the *install_apache.yml* and change the name of the package to any name that will not be recognized. What is the output?



```
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] **********************************************************
ok: [127.0.0.1]

TASK [update repository index] **************************************************
fatal: [127.0.0.1]: FAILED! => {"changed": false, "msg": "Failed to update apt cache: E:Release file for http://ph.archive.ubuntu.co
m/ubuntu/dists/jammy-updates/InRelease is not valid yet (invalid for another 18h 35min 5s). Updates for this repository will not be
applied., E:Release file for http://ph.archive.ubuntu.com/ubuntu/dists/jammy-backports/InRelease is not valid yet (invalid for anoth
er 17h 46min 2s). Updates for this repository will not be applied., E:Release file for http://security.ubuntu.com/ubuntu/dists/jammy
-security/InRelease is not valid yet (invalid for another 21h 8min 4s). Updates for this repository will not be applied., E:Release
file for http://archive.ubuntu.com/ubuntu/dists/jammy-updates/InRelease is not valid yet (invalid for another 21h 8min 30s). Updates
 for this repository will not be applied."}

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

bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$
```

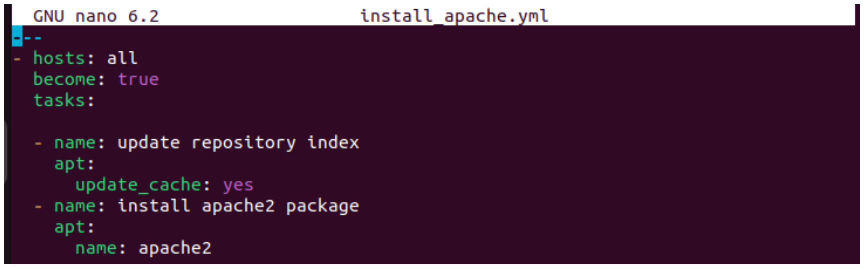
A fatal occurred and it fails to update the apache.

5. This time, we are going to put additional task to our playbook. Edit the *install_apache.yml*. As you can see, we are now adding an additional command, which is the *update_cache.* This command updates existing package-indexes on a supporting distro but not upgrading installed-packages (utilities) that were being installed.



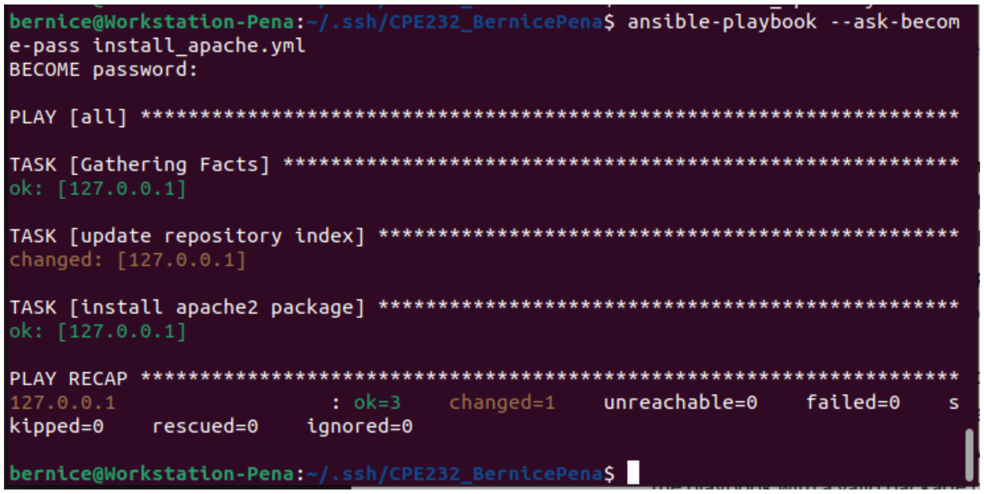
```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes

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

Save the changes to this file and exit.



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

  - name: update repository index
    apt:
      update_cache: yes
  - name: install apache2 package
    apt:
      name: apache2
```

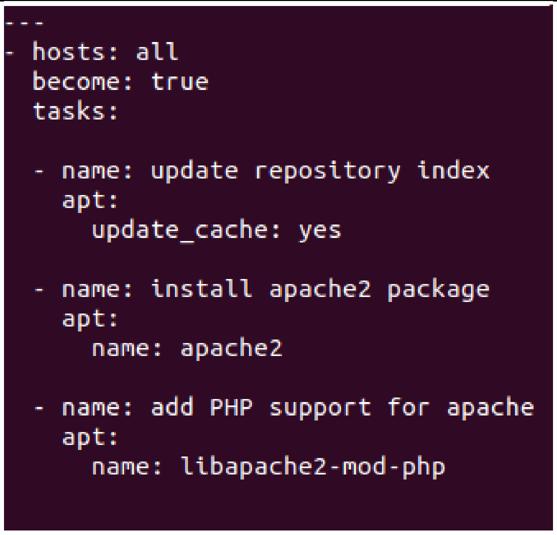
6. Run the playbook and describe the output. Did the new command change anything on the remote servers?



```
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ ansible-playbook --ask-becom
e-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************************
ok: [127.0.0.1]

TASK [update repository index] ***************************************************
changed: [127.0.0.1]

TASK [install apache2 package] ***************************************************
ok: [127.0.0.1]

PLAY RECAP ***********************************************************************
127.0.0.1                  : ok=3    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ 
```

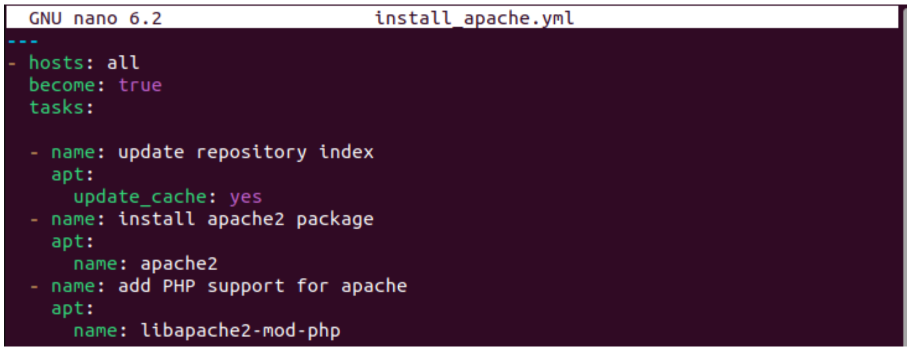
Yes, it seems that it changed something on the remote server. It updated the package repository index and it also installed the Apache web server package. These changes were reflected in the "changed" status in the output.

7. Edit again the *install_apache.yml*. This time, we are going to add a PHP support for the apache package we installed earlier.



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

  - name: update repository index
    apt:
      update_cache: yes

  - name: install apache2 package
    apt:
      name: apache2

  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
```

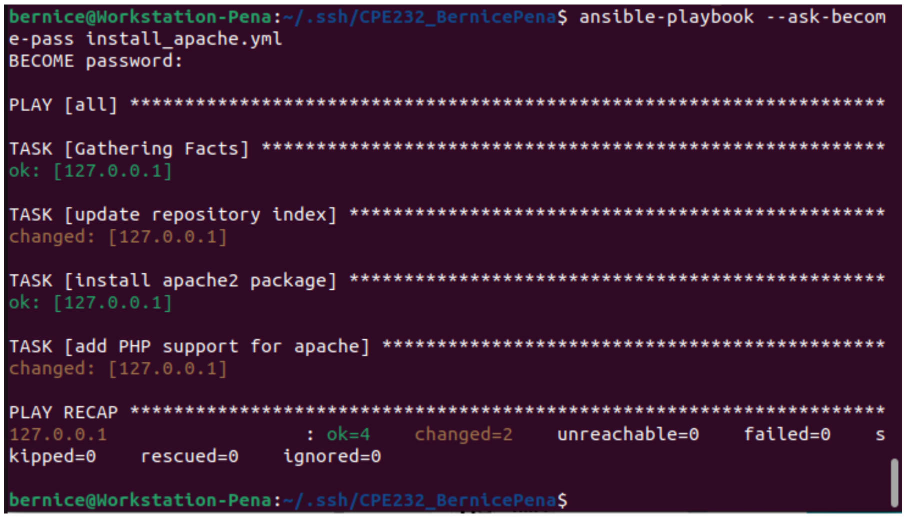
Save the changes to this file and exit.



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

  - name: update repository index
    apt:
      update_cache: yes
  - name: install apache2 package
    apt:
      name: apache2
  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
```

It changed something on the remote servers. In addition to updating the package repository index and installing the Apache web server, it also added PHP support for Apache.

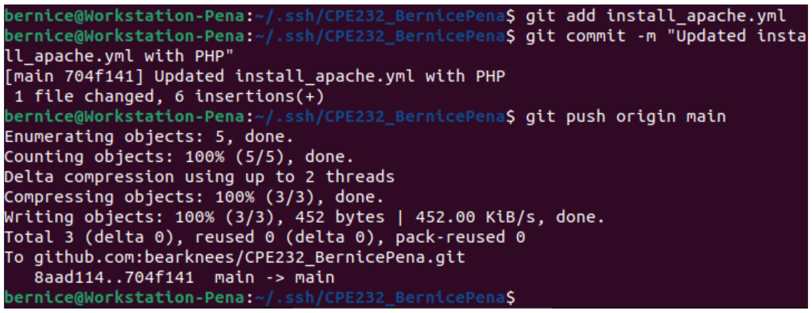
8. Run the playbook and describe the output. Did the new command change anything on the remote servers?



```
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ ansible-playbook --ask-becom
e-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [127.0.0.1]

TASK [update repository index] *************************************************
changed: [127.0.0.1]

TASK [install apache2 package] *************************************************
ok: [127.0.0.1]

TASK [add PHP support for apache] **********************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************
127.0.0.1                  : ok=4    changed=2    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$
```

It changed something on the remote servers. In addition to updating the package repository index and installing the Apache web server, it also added PHP support for Apache.

9. Finally, make sure that we are in sync with GitHub. Provide the link of your GitHub repository.



```
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ git add install_apache.yml
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ git commit -m "Updated insta
ll_apache.yml with PHP"
[main 704f141] Updated install_apache.yml with PHP
 1 file changed, 6 insertions(+)
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 452 bytes | 452.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:bearknees/CPE232_BernicePena.git
   8aad114..704f141  main -> main
bernice@Workstation-Pena:~/.ssh/CPE232_BernicePena$
```

https://github.com/bearknees/CPE232_BernicePena/blob/main/install_apache.yml

Reflections:

Answer the following:

1. What is the importance of using a playbook?
   Playbooks are tools that help us organize and structure automation tasks and workflows. They play a role in configuring installing and updating software packages, managing server settings and performing system administration tasks across servers simultaneously. By providing an easily understandable format, playbooks make automation more manageable, allow for efficient maintenance and scalability of our automation efforts. Also, they ensure consistency and reliability in server configurations and software installations while enabling collaboration through version control and change tracking. Ultimately, playbooks are essential for streamlining tasks enhancing efficiency and maintaining infrastructure as code practices.

2. Summarize what we have done on this activity.
   Through this activity, I learned how to harness Ansible for managing remote systems via automation. Starting with ad-hoc command execution to keep package details current and install software including vim and snapd on remote systems, I then developed an installation playbook for the Apache web server that included extra PHP capabilities. How did the execution of the playbook change the state of remote servers? By doing so, the importance of leveraging playbooks in infrastructure management was

included and I was able to learn about it, and changes were synchronized with a GitHub repository.