Name: Sonny Jay B. Bencito	**Date Performed: 09/11/2022**
Course/Section: CPE32S24	**Date Submitted: 09/13/2022**
Instructor: Dr. Jonathan Taylar	**Semester and SY: 1st sem – 2022-2023**

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

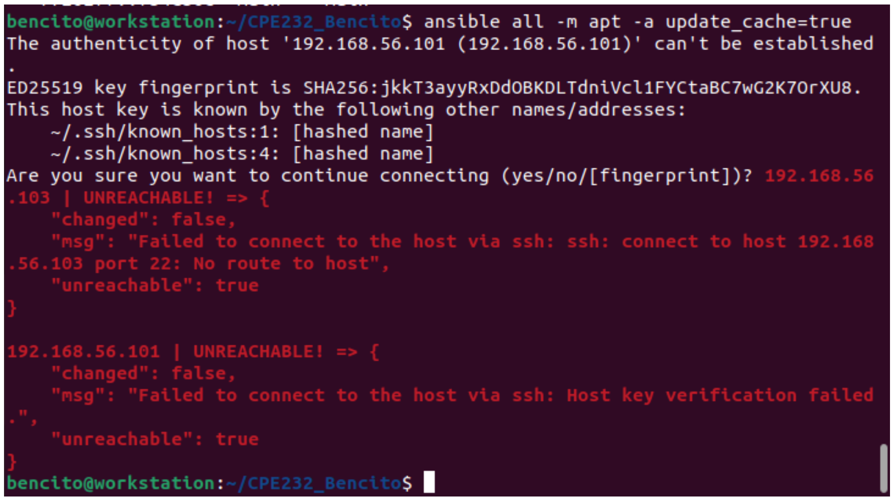
So far, we have not performed ansible commands that makes changes to the remote servers. We manage to gather facts and connect to the remote machines, but we still did not make changes on those machines. In this activity, we will learn to use commands that would install, update, and upgrade packages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



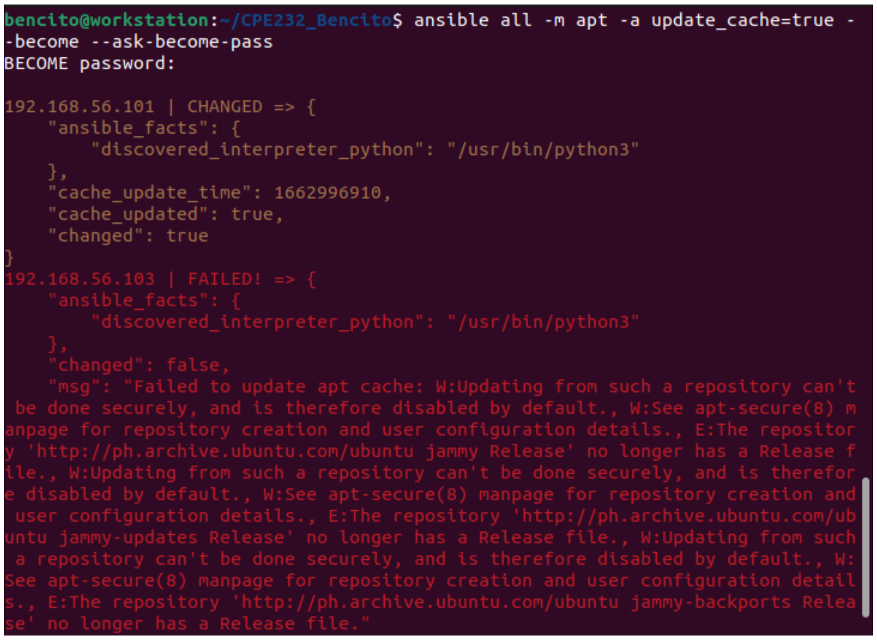
```
bencito@workstation:~/CPE232_Bencito$ ansible all -m apt -a update_cache=true
The authenticity of host '192.168.56.101 (192.168.56.101)' can't be established
.
ED25519 key fingerprint is SHA256:jkkT3ayyRxDdOBKDLTdniVcl1FYCtaBC7wG2K7OrXU8.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
    ~/.ssh/known_hosts:4: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? 192.168.56
.103 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168
.56.103 port 22: No route to host",
    "unreachable": true
}

192.168.56.101 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: Host key verification failed
.",
    "unreachable": true
}
bencito@workstation:~/CPE232_Bencito$
```

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

It's not successful because it's not unreachable

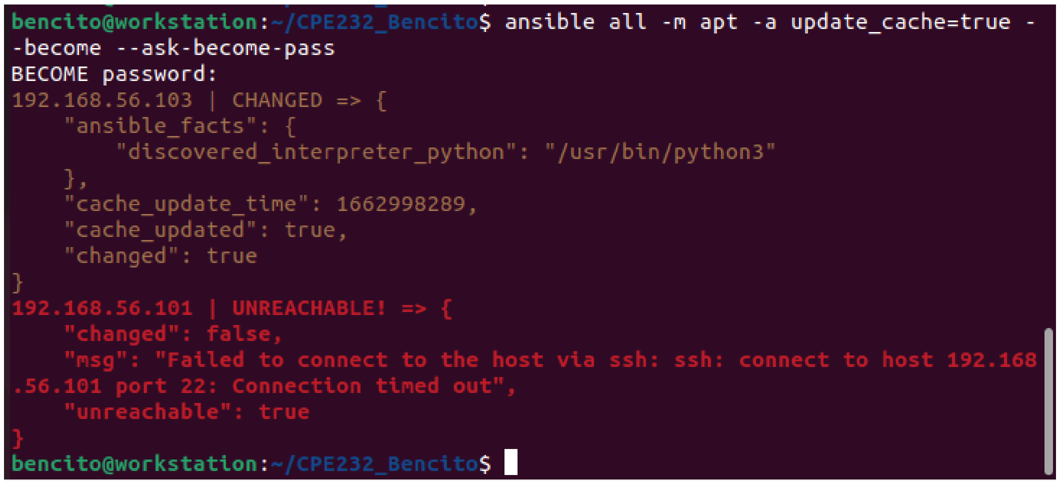
Try editing the command and add something that would elevate the privilege. Issue the command *ansible all -m apt -a update_cache=true --become --ask-become-pass.* Enter the sudo password when prompted. You will notice now that the output of this command is a success. The *update_cache=true* is the same 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.

<u>Only Server 1 open and Updated</u>



```
bencito@workstation:~/CPE232_Bencito$ ansible all -m apt -a update_cache=true -
-become --ask-become-pass
BECOME password:

192.168.56.101 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662996910,
    "cache_updated": true,
    "changed": true
}
192.168.56.103 | FAILED! => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "msg": "Failed to update apt cache: W:Updating from such a repository can't
 be done securely, and is therefore disabled by default., W:See apt-secure(8) m
anpage for repository creation and user configuration details., E:The repositor
y 'http://ph.archive.ubuntu.com/ubuntu jammy Release' no longer has a Release f
ile., W:Updating from such a repository can't be done securely, and is therefor
e disabled by default., W:See apt-secure(8) manpage for repository creation and
 user configuration details., E:The repository 'http://ph.archive.ubuntu.com/ub
untu jammy-updates Release' no longer has a Release file., W:Updating from such
 a repository can't be done securely, and is therefore disabled by default., W:
See apt-secure(8) manpage for repository creation and user configuration detail
s., E:The repository 'http://ph.archive.ubuntu.com/ubuntu jammy-backports Relea
se' no longer has a Release file."
```

Only server 2 open



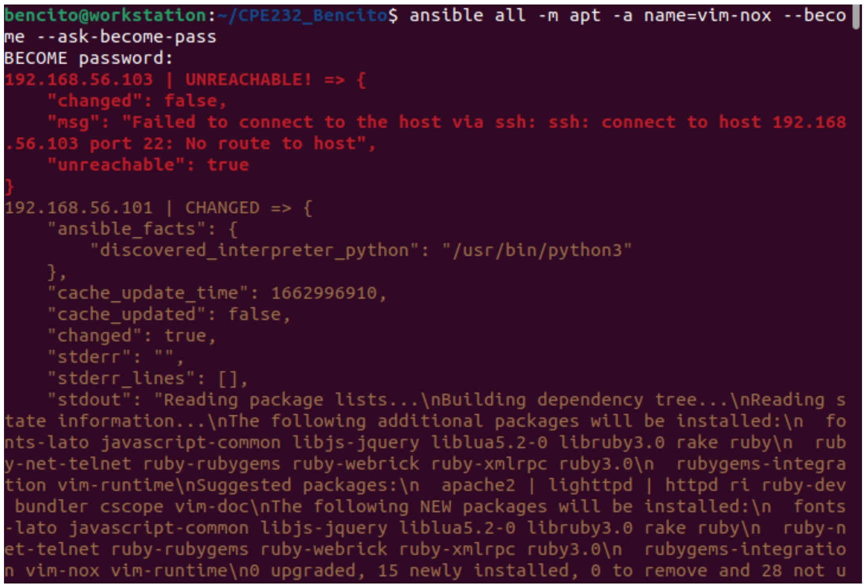
```
bencito@workstation:~/CPE232_Bencito$ ansible all -m apt -a update_cache=true -
-become --ask-become-pass
BECOME password:
192.168.56.103 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662998289,
    "cache_updated": true,
    "changed": true
}
192.168.56.101 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168
.56.101 port 22: Connection timed out",
    "unreachable": true
}
bencito@workstation:~/CPE232_Bencito$
```

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

Yes, the first is not working.

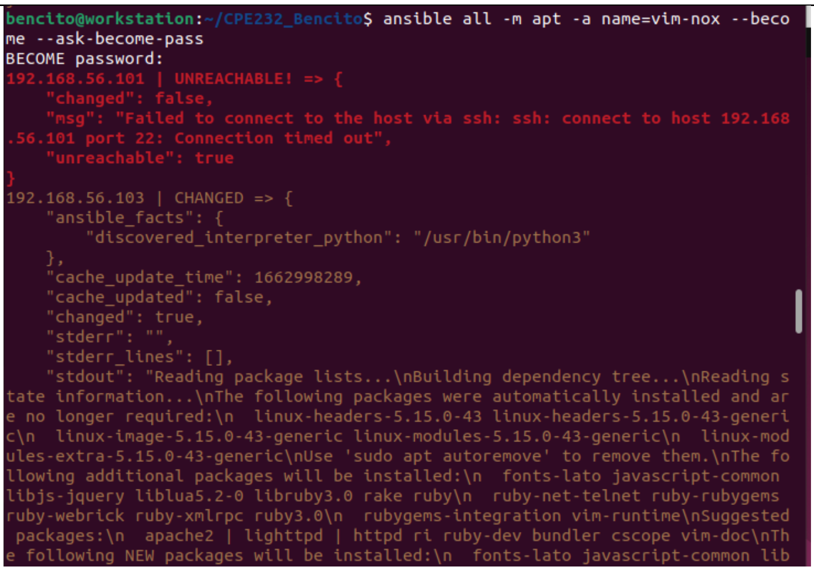
2. Let's try to install VIM, which is an almost compatible version of the UNIX editor Vi. To do this, we will just changed the module part in 1.1 instruction. Here is the command: *ansible all -m apt -a name=vim-nox --become --ask-become-pass.* The command would take some time after typing the password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

Server 1 only



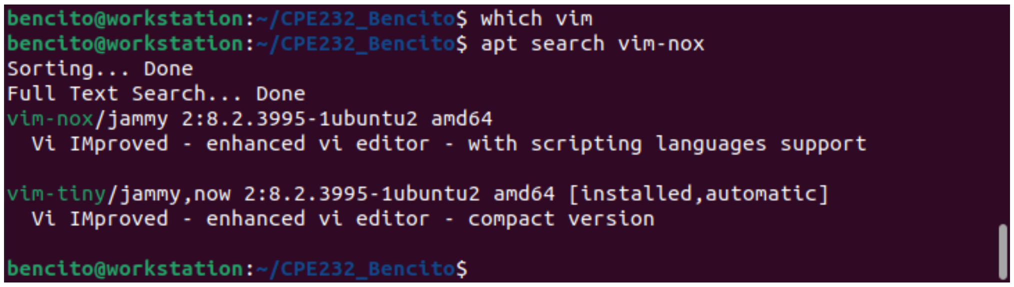
```
bencito@workstation:~/CPE232_Bencito$ ansible all -m apt -a name=vim-nox --beco
me --ask-become-pass
BECOME password:
192.168.56.103 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168
.56.103 port 22: No route to host",
    "unreachable": true
}
192.168.56.101 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662996910,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading s
tate information...\nThe following additional packages will be installed:\n  fo
nts-lato javascript-common libjs-jquery liblua5.2-0 libruby3.0 rake ruby\n  rub
y-net-telnet ruby-rubygems ruby-webrick ruby-xmlrpc ruby3.0\n  rubygems-integra
tion vim-runtime\nSuggested packages:\n  apache2 | lighttpd | httpd ri ruby-dev
 bundler cscope vim-doc\nThe following NEW packages will be installed:\n  fonts
-lato javascript-common libjs-jquery liblua5.2-0 libruby3.0 rake ruby\n  ruby-n
et-telnet ruby-rubygems ruby-webrick ruby-xmlrpc ruby3.0\n  rubygems-integratio
n vim-nox vim-runtime\n0 upgraded, 15 newly installed, 0 to remove and 28 not u
```

Server 2 only



```
bencito@workstation:~/CPE232_Bencito$ ansible all -m apt -a name=vim-nox --beco
me --ask-become-pass
BECOME password:
192.168.56.101 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168
.56.101 port 22: Connection timed out",
    "unreachable": true
}
192.168.56.103 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662998289,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading s
tate information...\nThe following packages were automatically installed and ar
e no longer required:\n  linux-headers-5.15.0-43 linux-headers-5.15.0-43-generi
c\n  linux-image-5.15.0-43-generic linux-modules-5.15.0-43-generic\n  linux-mod
ules-extra-5.15.0-43-generic\nUse 'sudo apt autoremove' to remove them.\nThe fo
llowing additional packages will be installed:\n  fonts-lato javascript-common
libjs-jquery liblua5.2-0 libruby3.0 rake ruby\n  ruby-net-telnet ruby-rubygems
ruby-webrick ruby-xmlrpc ruby3.0\n  rubygems-integration vim-runtime\nSuggested
 packages:\n  apache2 | lighttpd | httpd ri ruby-dev bundler cscope vim-doc\nTh
e following NEW packages will be installed:\n  fonts-lato javascript-common lib
```

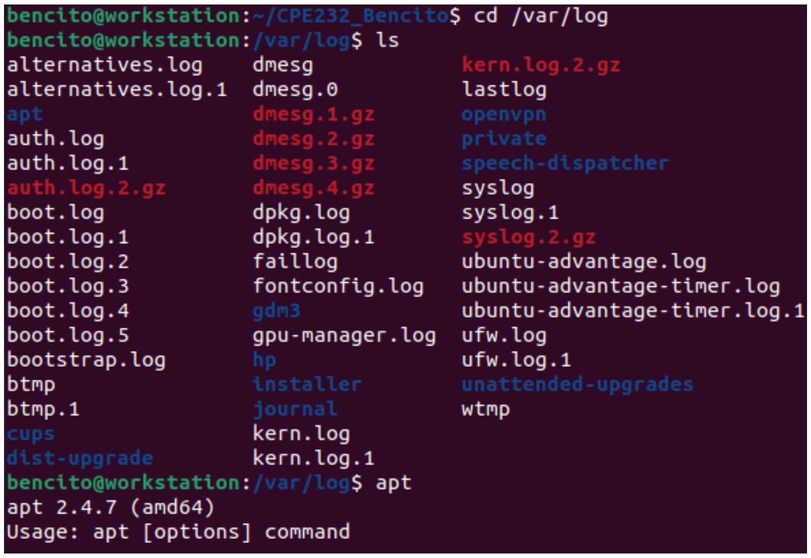
2.1 Verify that you have installed the package in the remote servers. Issue the command *which vim* and the command *apt search vim-nox* respectively. Was the command successful?



```
bencito@workstation:~/CPE232_Bencito$ which vim
bencito@workstation:~/CPE232_Bencito$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/jammy 2:8.2.3995-1ubuntu2 amd64
  Vi IMproved - enhanced vi editor - with scripting languages support

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

bencito@workstation:~/CPE232_Bencito$
```

Yes, It's successful

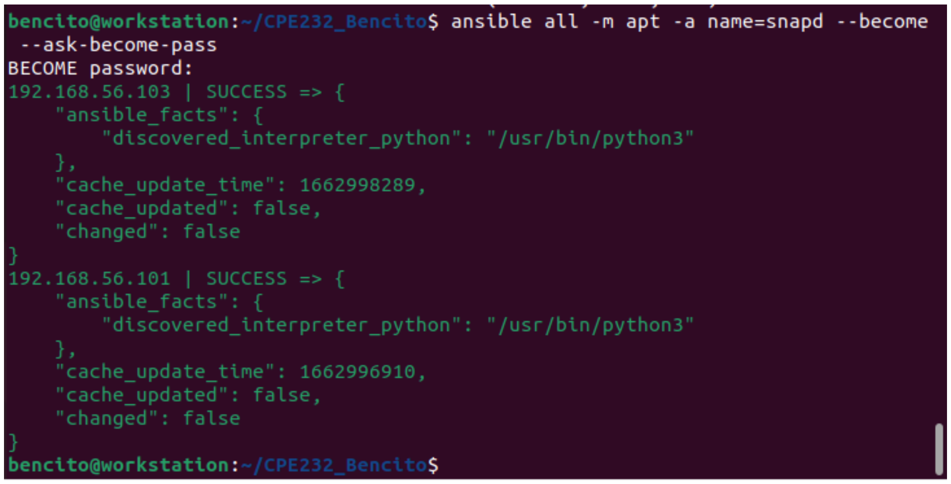
2.2 Check the logs in the servers using the following commands: *cd /var/log*. After this, issue the command *ls,* go to the folder *apt* and open history.log. Describe what you see in the history.log.



```
bencito@workstation:~/CPE232_Bencito$ cd /var/log
bencito@workstation:/var/log$ ls
alternatives.log    dmesg              kern.log.2.gz
alternatives.log.1  dmesg.0            lastlog
apt                 dmesg.1.gz         openvpn
auth.log            dmesg.2.gz         private
auth.log.1          dmesg.3.gz         speech-dispatcher
auth.log.2.gz       dmesg.4.gz         syslog
boot.log            dpkg.log           syslog.1
boot.log.1          dpkg.log.1         syslog.2.gz
boot.log.2          faillog            ubuntu-advantage.log
boot.log.3          fontconfig.log     ubuntu-advantage-timer.log
boot.log.4          gdm3               ubuntu-advantage-timer.log.1
boot.log.5          gpu-manager.log    ufw.log
bootstrap.log       hp                 ufw.log.1
btmp                installer          unattended-upgrades
btmp.1              journal            wtmp
cups                kern.log
dist-upgrade        kern.log.1
bencito@workstation:/var/log$ apt
apt 2.4.7 (amd64)
Usage: apt [options] command
```

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*

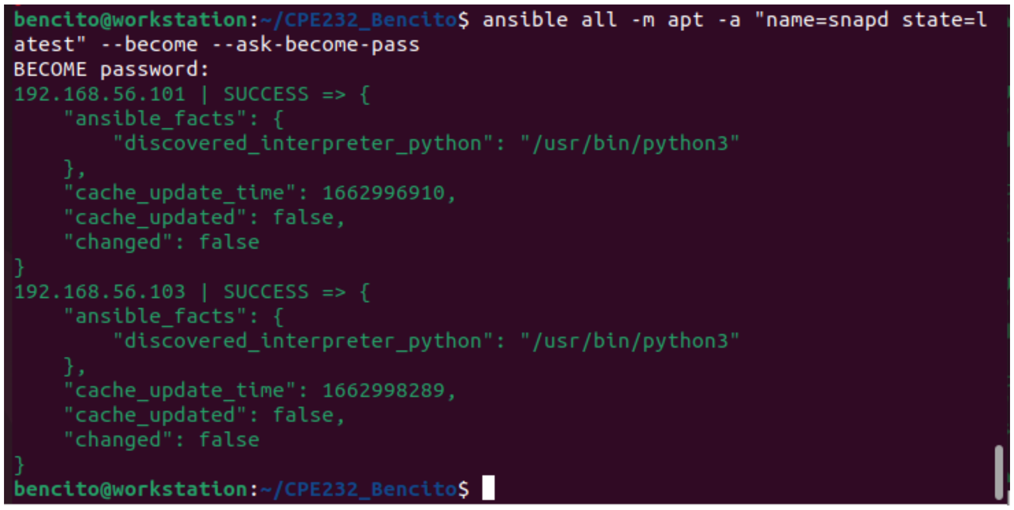


```
bencito@workstation:~/CPE232_Bencito$ ansible all -m apt -a name=snapd --become
 --ask-become-pass
BECOME password:
192.168.56.103 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662998289,
    "cache_updated": false,
    "changed": false
}
192.168.56.101 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662996910,
    "cache_updated": false,
    "changed": false
}
bencito@workstation:~/CPE232_Bencito$
```

   Can you describe the result of this command? Is it a success? Did it change anything in the remote servers?

   The command was successful but there is no any changes.

   3.2 Now, try to issue this command: *ansible all -m apt -a "name=snapd state=latest" --become --ask-become-pass*

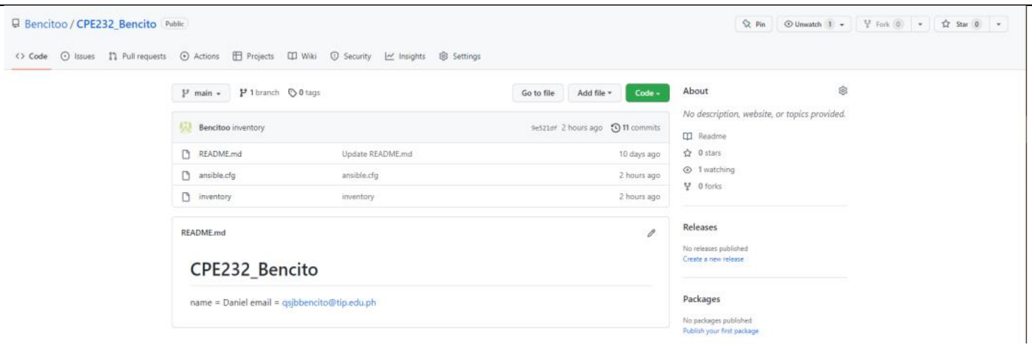


```
bencito@workstation:~/CPE232_Bencito$ ansible all -m apt -a "name=snapd state=l
atest" --become --ask-become-pass
BECOME password:
192.168.56.101 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662996910,
    "cache_updated": false,
    "changed": false
}
192.168.56.103 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662998289,
    "cache_updated": false,
    "changed": false
}
bencito@workstation:~/CPE232_Bencito$
```

   Describe the output of this command. Notice how we added the command *state=latest* and placed them in double quotations.

   It's just the same, it just check if the snapd is updated.

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



Bencitoo / CPE232_Bencito Public

Bencitoo inventory	9e521df 2 hours ago	11 commits
README.md	Update README.md	10 days ago
ansible.cfg	ansible.cfg	2 hours ago
inventory	inventory	2 hours ago

README.md

CPE232_Bencito

name = Daniel email = qsjbbencito@tip.edu.ph

About

No description, website, or topics provided.

Readme
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

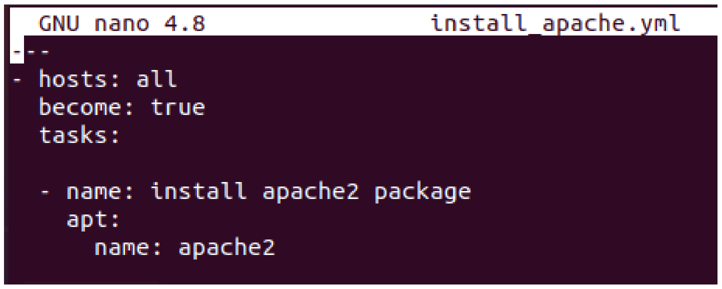
Packages

No packages published
Publish your first package

Task 2: Writing our First Playbook

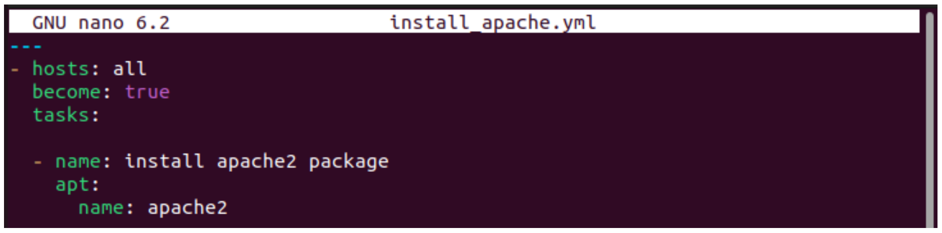
1. With ad hoc commands, we can simplify the administration of remote servers. For example, we can install updates, packages, and applications, etc. However, the real strength of ansible comes from its playbooks. When we write a playbook, we can define the state that we want our servers to be in and 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
```

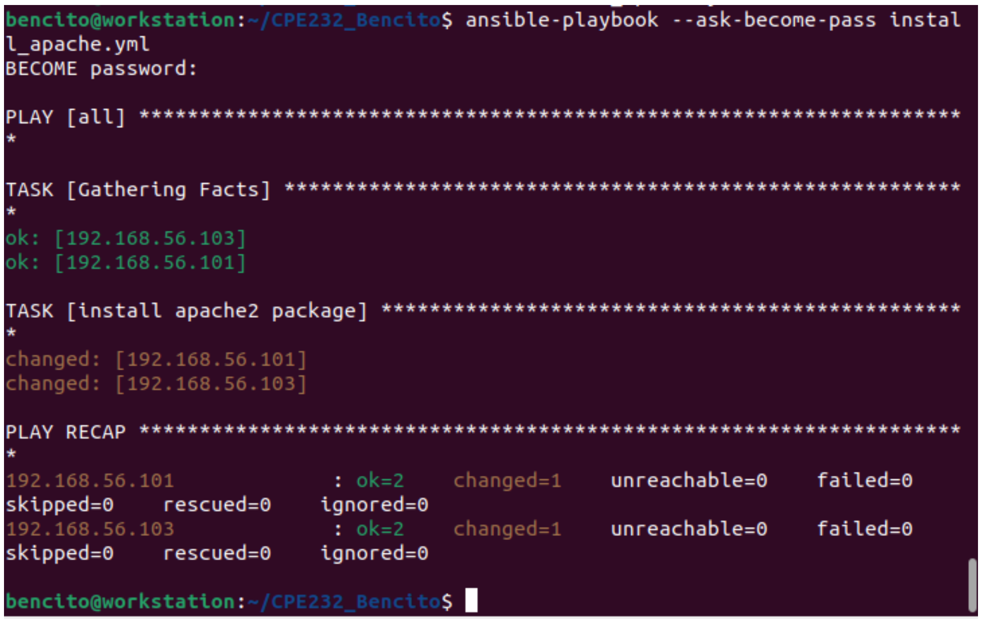
   Screenshot



```
 GNU nano 6.2                    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.

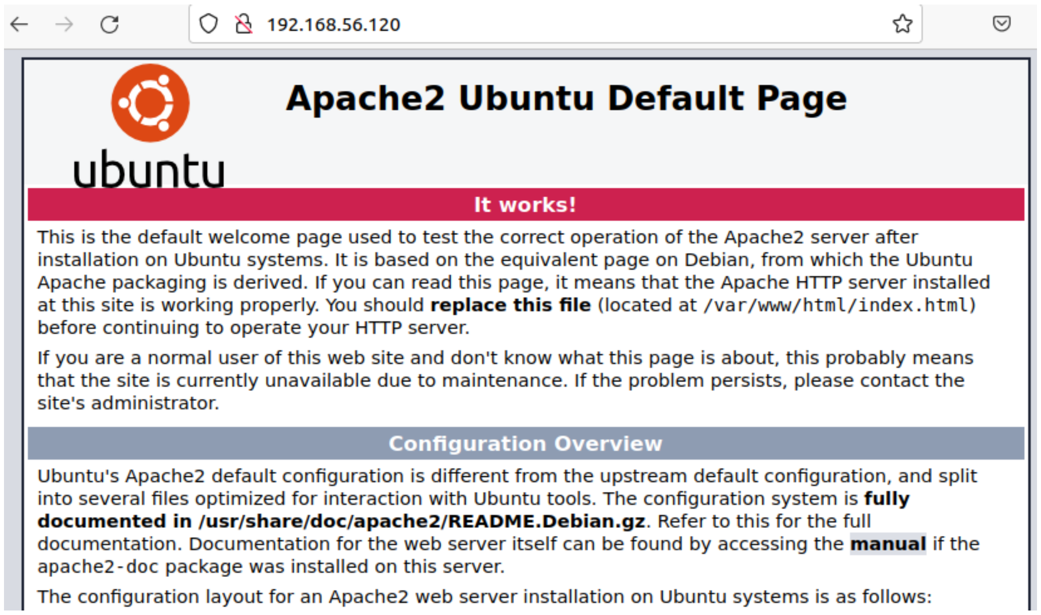
2. Run the yml file using the command: *ansible-playbook --ask-become-pass install_apache.yml.* Describe the result of this command.



```
bencito@workstation:~/CPE232_Bencito$ ansible-playbook --ask-become-pass instal
l_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.103]
ok: [192.168.56.101]

TASK [install apache2 package] *************************************************
*
changed: [192.168.56.101]
changed: [192.168.56.103]

PLAY RECAP *********************************************************************
*
192.168.56.101             : ok=2    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=2    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

bencito@workstation:~/CPE232_Bencito$
```

It successfully installed and there is no error found.

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

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.

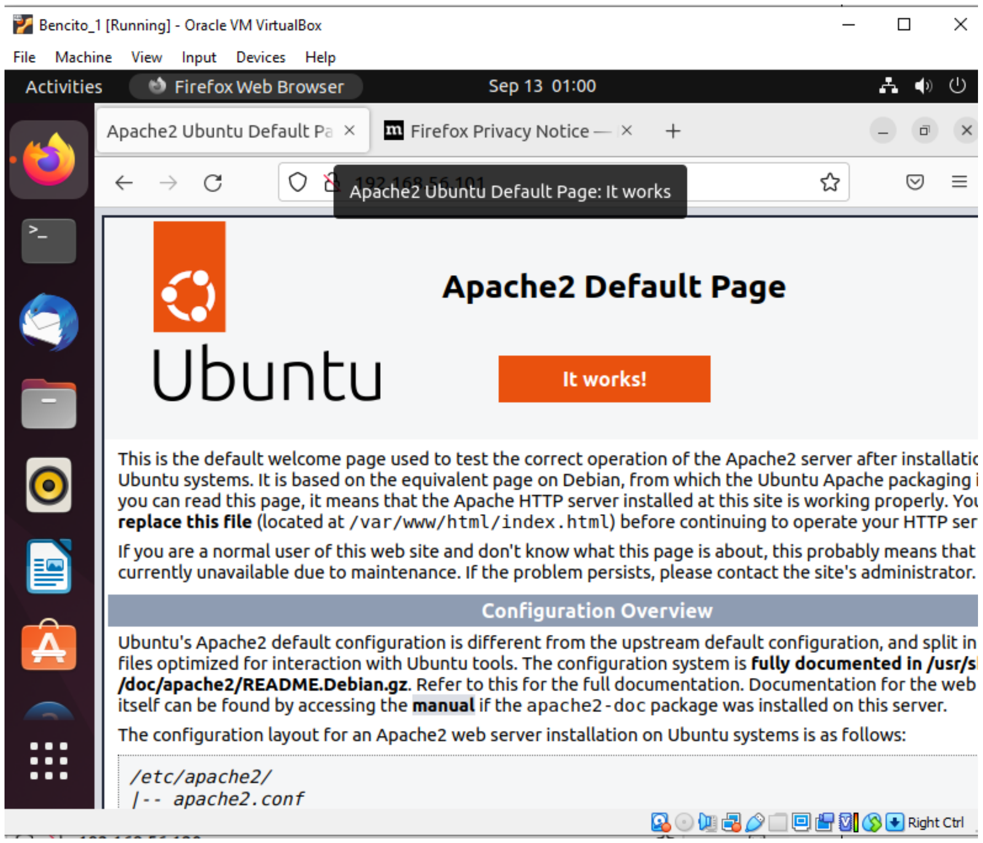
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

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:

Server 1


Bencito_1 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Firefox Web Browser Sep 13 01:00

Apache2 Ubuntu Default Pa Firefox Privacy Notice —

Apache2 Ubuntu Default Page: It works

Apache2 Default Page

Ubuntu

It works!

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

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

Configuration Overview

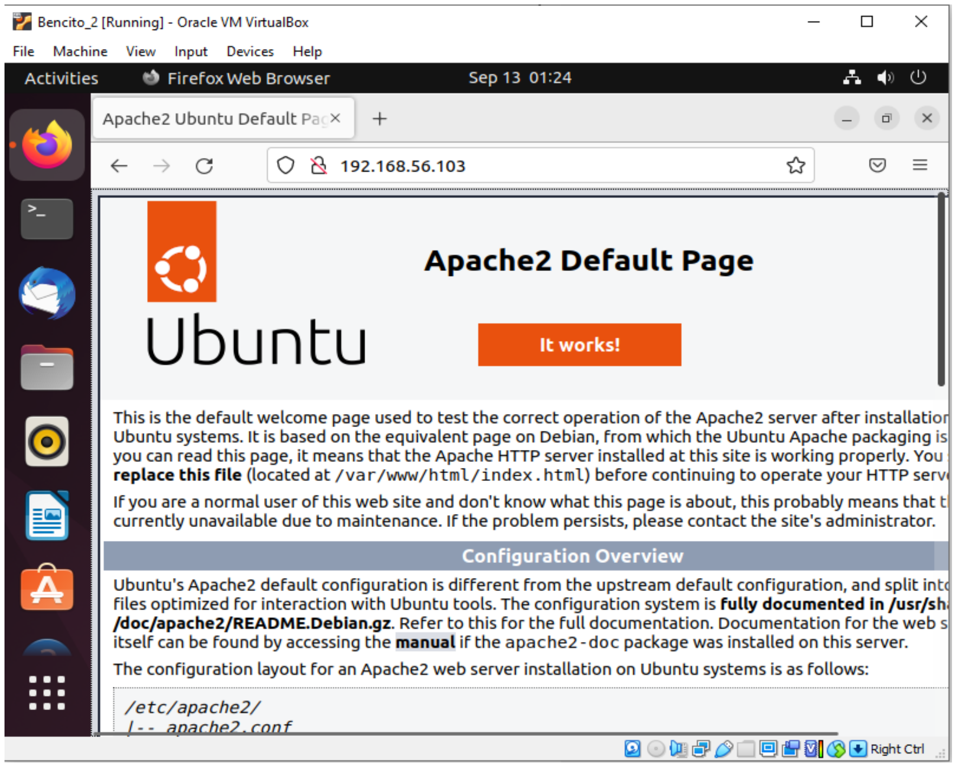
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split in
files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/s**
/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web
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
```

Right Ctrl

Server 2


Bencito_2 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Firefox Web Browser Sep 13 01:24

Apache2 Ubuntu Default Pa

192.168.56.103

Apache2 Default Page

Ubuntu

It works!

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

If you are a normal user of this web site and don't know what this page is about, this probably means that t
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
files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/sh**
/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web s
itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

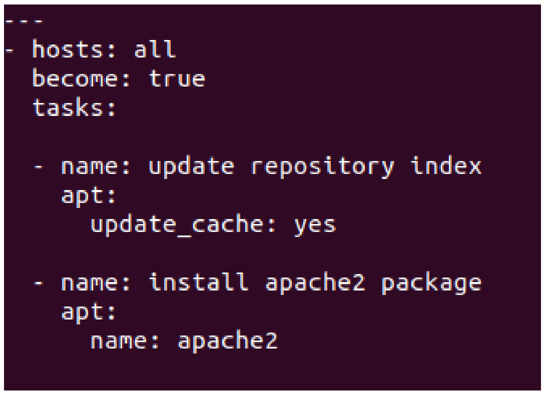
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
```

Right Ctrl

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?
   The out will be error. Because the package will not read it.

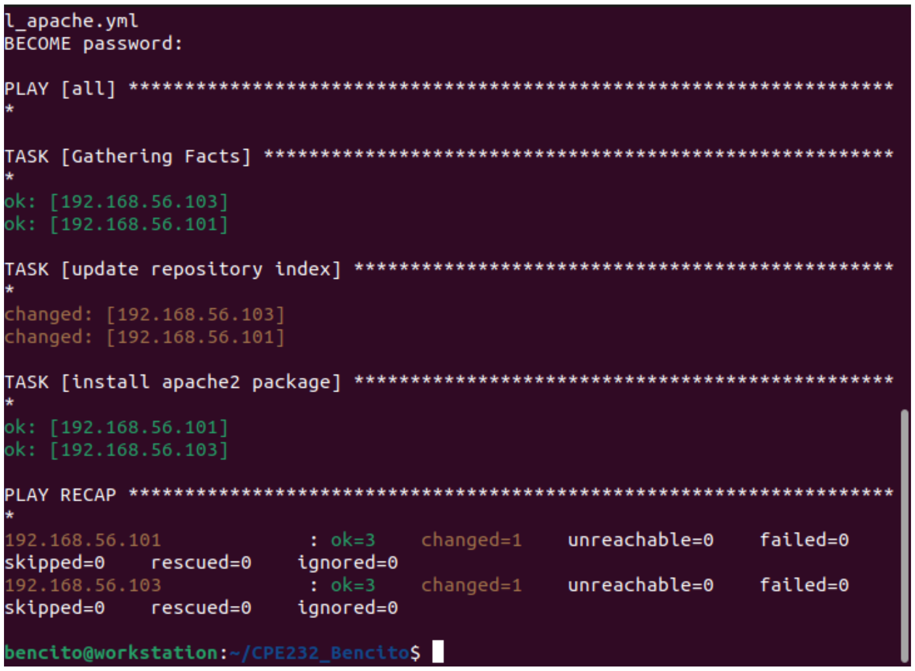
5. This time, we are going to put additional task to our playbook. Edit the *install_apache.yml*. As you can see, we are now adding an additional command, which is the *update_cache.* This command updates existing package-indexes on a supporting distro but not upgrading installed-packages (utilities)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

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



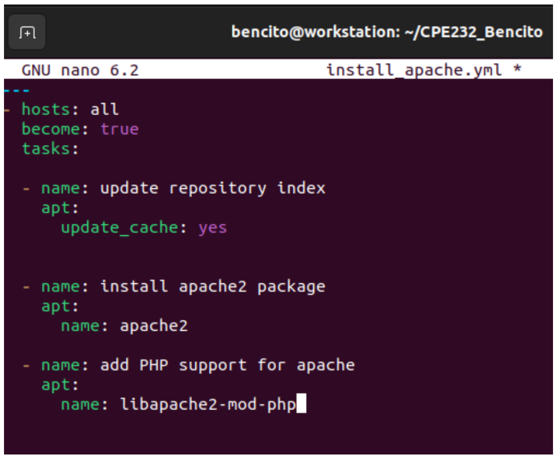
```
l_apache.yml
BECOME password:

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

TASK [Gathering Facts] ******************************************************
*
ok: [192.168.56.103]
ok: [192.168.56.101]

TASK [update repository index] **********************************************
*
changed: [192.168.56.103]
changed: [192.168.56.101]

TASK [install apache2 package] **********************************************
*
ok: [192.168.56.101]
ok: [192.168.56.103]

PLAY RECAP ******************************************************************
*
192.168.56.101             : ok=3    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=3    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

bencito@workstation:~/CPE232_Bencito$ 
```

   There are changed on the repository index.

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

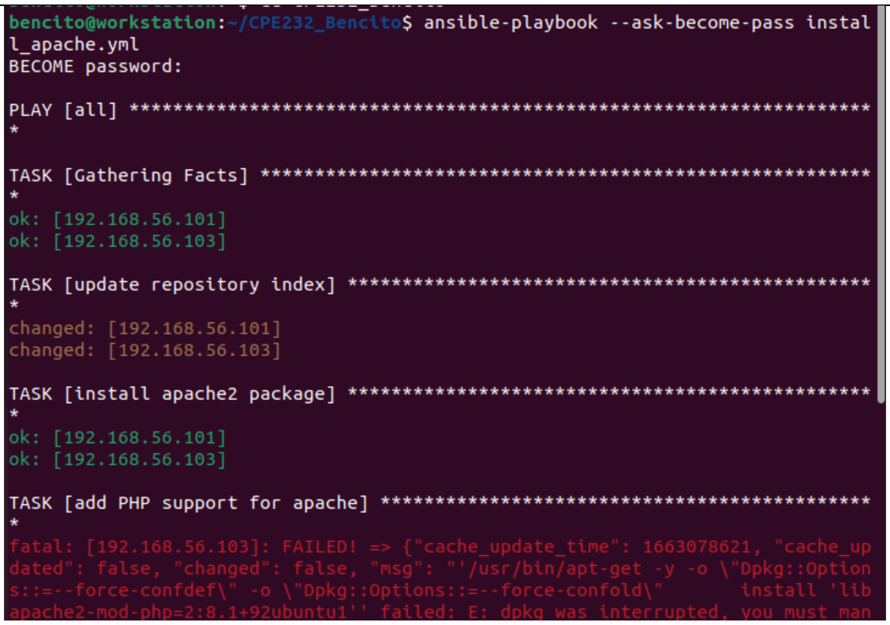
```
---
- hosts: all
  become: true
  tasks:

  - 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.

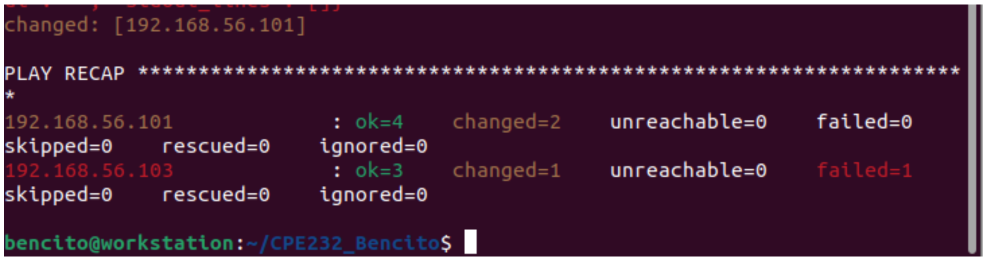


```
bencito@workstation: ~/CPE232_Bencito
  GNU nano 6.2                          install_apache.yml *
---
- hosts: all
  become: true
  tasks:

  - 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
```

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



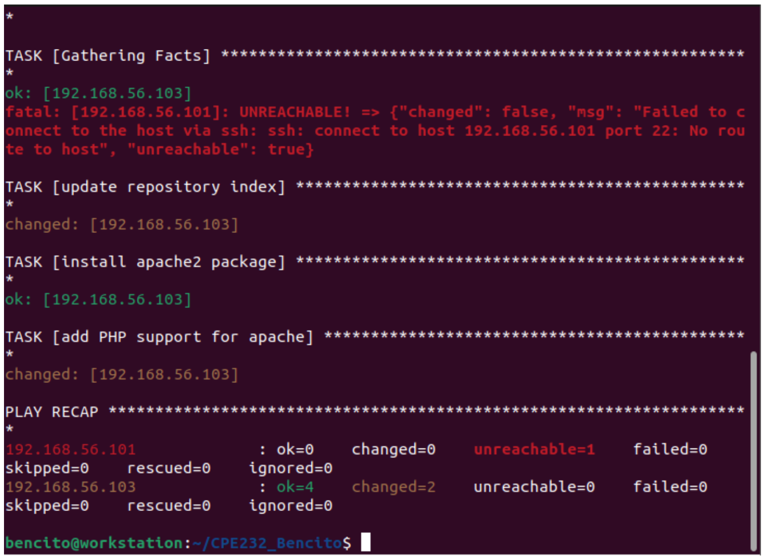
```
bencito@workstation:~/CPE232_Bencito$ ansible-playbook --ask-become-pass instal
l_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.101]
ok: [192.168.56.103]

TASK [update repository index] *************************************************
*
changed: [192.168.56.101]
changed: [192.168.56.103]

TASK [install apache2 package] *************************************************
*
ok: [192.168.56.101]
ok: [192.168.56.103]

TASK [add PHP support for apache] **********************************************
*
fatal: [192.168.56.103]: FAILED! => {"cache_update_time": 1663078621, "cache_up
dated": false, "changed": false, "msg": "'/usr/bin/apt-get -y -o \"Dpkg::Option
s::=--force-confdef\" -o \"Dpkg::Options::=--force-confold\"       install 'lib
apache2-mod-php=2:8.1+92ubuntu1'' failed: E: dpkg was interrupted, you must man
```



```
changed: [192.168.56.101]

PLAY RECAP *********************************************************************
*
192.168.56.101             : ok=4    changed=2    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=3    changed=1    unreachable=0    failed=1
skipped=0    rescued=0    ignored=0

bencito@workstation:~/CPE232_Bencito$
```

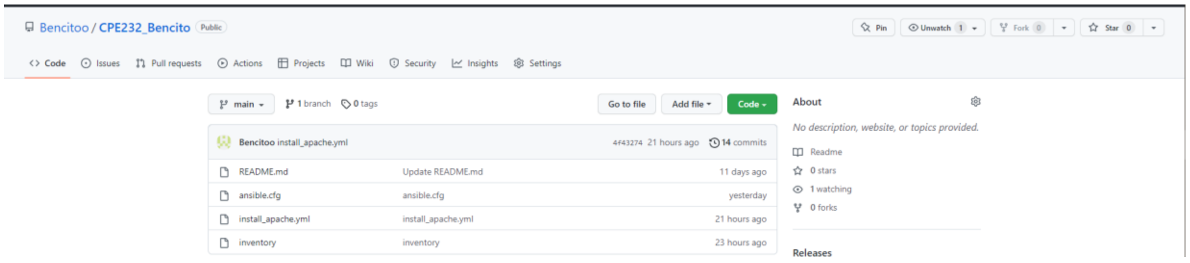
<u>They have a one fail changes on the server 2. Because you need first to reconfigure it. On the server 1 all changes will be assigned without an error.</u>

Server 2 fixed



```
*

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.103]
fatal: [192.168.56.101]: UNREACHABLE! => {"changed": false, "msg": "Failed to c
onnect to the host via ssh: ssh: connect to host 192.168.56.101 port 22: No rou
te to host", "unreachable": true}

TASK [update repository index] *************************************************
*
changed: [192.168.56.103]

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

TASK [add PHP support for apache] **********************************************
*
changed: [192.168.56.103]

PLAY RECAP *********************************************************************
*
192.168.56.101             : ok=0    changed=0    unreachable=1    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=4    changed=2    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

bencito@workstation:~/CPE232_Bencito$
```

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



Bencitoo / CPE232_Bencito Public

Bencitoo install_apache.yml 4f43274 21 hours ago 14 commits

README.md	Update README.md	11 days ago
ansible.cfg	ansible.cfg	yesterday
install_apache.yml	install_apache.yml	21 hours ago
inventory	inventory	23 hours ago

About

No description, website, or topics provided.

Readme
0 stars
1 watching
0 forks

Releases

Link: https://github.com/Bencitoo/CPE232_Bencito

Reflections:

Answer the following:

1. What is the importance of using a playbook?

   The importance of the playbook was to make fastest the work on any server. For example, you don't need to install a software manually one by one on the pc. You just need to open it and connect to each other so that the you need to just create a playbook to install it.

2. Summarize what we have done on this activity.

   When I'm doing this activity. When I see the file, I don't know what I'm going to do because of the error that I have encounter. But you just only need to create a file on the repository that connects to the github and your 2 clone servers. After doing that I try to run again the given commands on the file and it was successful, I just follow it and until I go to the task 2 that you need to create a playbook. It was easy for me to create a playbook because the given code is given you just need to input it on your repository and need to open the two servers to work the command. I just follow it and I have a little bit error but you just need to reconfigure it and re update the servers.

Honor Pledge:

I affirm that I shall not give or receive any unauthorized help on this assignment and that all work shall be my own.