Name: John Renzo L. Mendoza	**Date Performed:** September 09, 2023
Course/Section: CPE31S5	**Date Submitted:** September 12, 2023
Instructor: Engr. Roman Richard	**Semester and SY:** 1st Semester 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

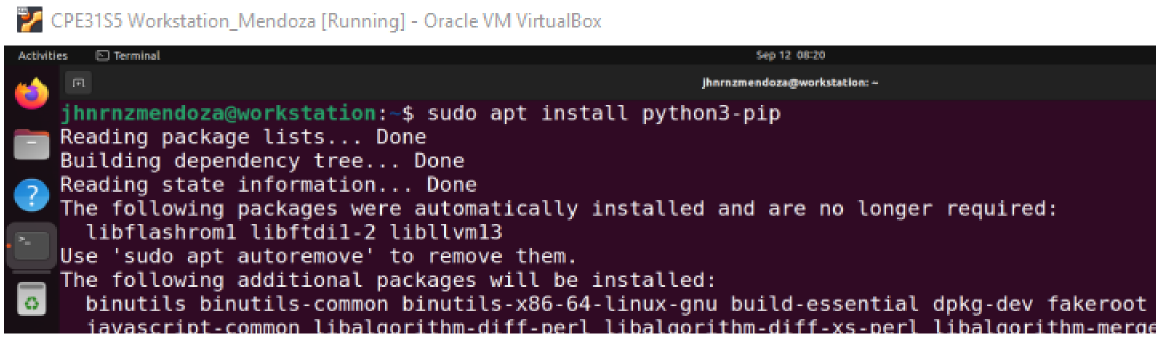
2. **Discussion:**

Elevated Ad hoc commands

So far, we have not performed ansible commands that make changes to the remote servers. We managed 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.

Installing python3 as it is needed in making playbooks with ansible.

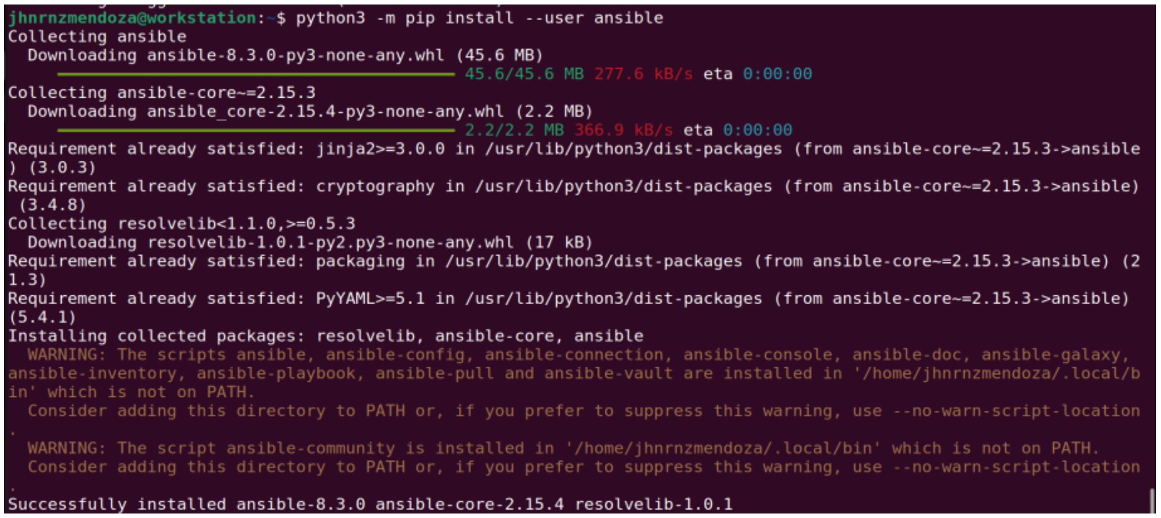


CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox

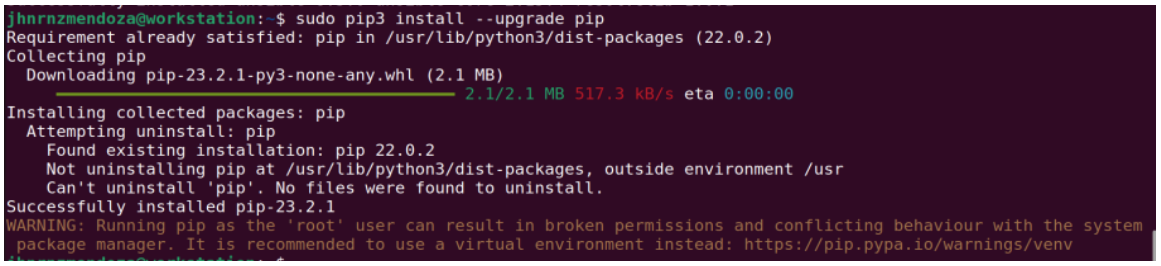
Activities Terminal Sep 12 08:20

jhnrnzmendoza@workstation: ~

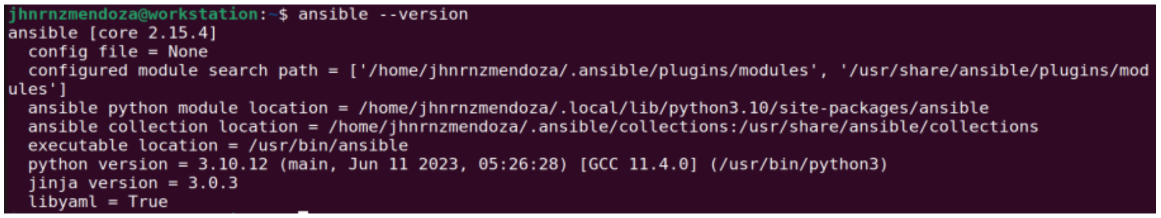
```
jhnrnzmendoza@workstation:~$ sudo apt install python3-pip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libflashrom1 libftdi1-2 libllvm13
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev fakeroot
  javascript-common libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge
```



```
jhnrnzmendoza@workstation:~$ python3 -m pip install --user ansible
Collecting ansible
  Downloading ansible-8.3.0-py3-none-any.whl (45.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.6/45.6 MB 277.6 kB/s eta 0:00:00
Collecting ansible-core~=2.15.3
  Downloading ansible_core-2.15.4-py3-none-any.whl (2.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.2/2.2 MB 366.9 kB/s eta 0:00:00
Requirement already satisfied: jinja2>=3.0.0 in /usr/lib/python3/dist-packages (from ansible-core~=2.15.3->ansible
) (3.0.3)
Requirement already satisfied: cryptography in /usr/lib/python3/dist-packages (from ansible-core~=2.15.3->ansible)
 (3.4.8)
Collecting resolvelib<1.1.0,>=0.5.3
  Downloading resolvelib-1.0.1-py2.py3-none-any.whl (17 kB)
Requirement already satisfied: packaging in /usr/lib/python3/dist-packages (from ansible-core~=2.15.3->ansible) (2
1.3)
Requirement already satisfied: PyYAML>=5.1 in /usr/lib/python3/dist-packages (from ansible-core~=2.15.3->ansible)
(5.4.1)
Installing collected packages: resolvelib, ansible-core, ansible
  WARNING: The scripts ansible, ansible-config, ansible-connection, ansible-console, ansible-doc, ansible-galaxy,
ansible-inventory, ansible-playbook, ansible-pull and ansible-vault are installed in '/home/jhnrnzmendoza/.local/b
in' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location
.
  WARNING: The script ansible-community is installed in '/home/jhnrnzmendoza/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location
.
Successfully installed ansible-8.3.0 ansible-core-2.15.4 resolvelib-1.0.1
```



```
jhnrnzmendoza@workstation:~$ sudo pip3 install --upgrade pip
Requirement already satisfied: pip in /usr/lib/python3/dist-packages (22.0.2)
Collecting pip
  Downloading pip-23.2.1-py3-none-any.whl (2.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.1/2.1 MB 517.3 kB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.0.2
    Not uninstalling pip at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'pip'. No files were found to uninstall.
Successfully installed pip-23.2.1
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system
 package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

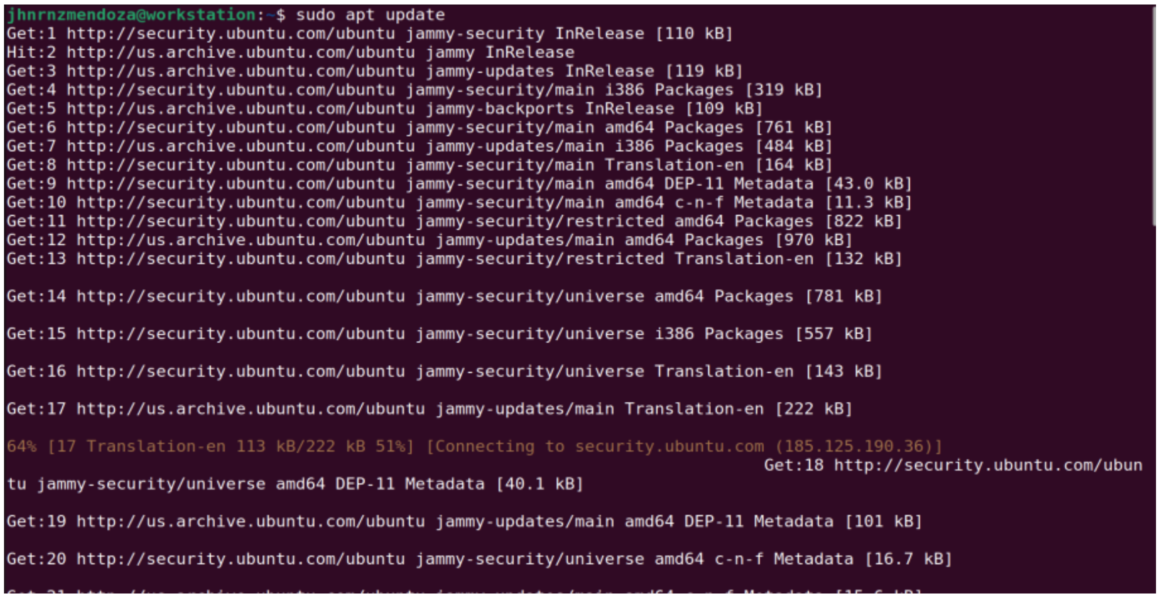


```
jhnrnzmendoza@workstation:~$ ansible --version
ansible [core 2.15.4]
  config file = None
  configured module search path = ['/home/jhnrnzmendoza/.ansible/plugins/modules', '/usr/share/ansible/plugins/mod
ules']
  ansible python module location = /home/jhnrnzmendoza/.local/lib/python3.10/site-packages/ansible
  ansible collection location = /home/jhnrnzmendoza/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
```

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 are often defined in /etc/apt/sources.list file and other files located in /etc/apt/sources.list.d/ directory. So, when you run the 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:

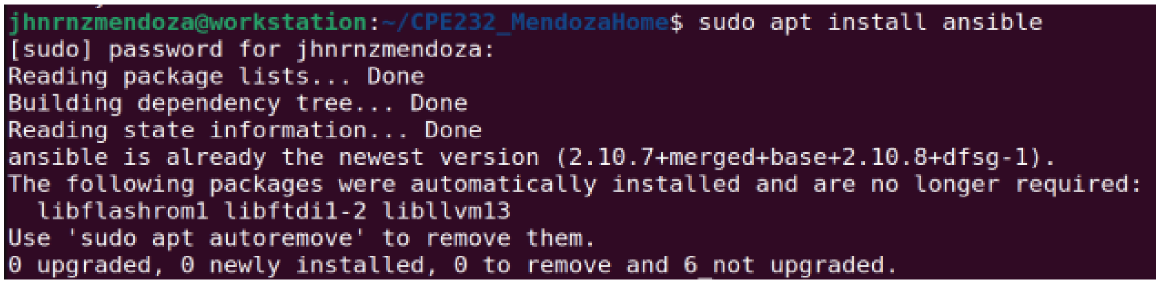
   Locally

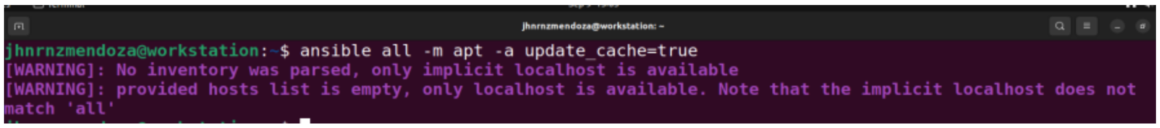
   ```
   jhnrnzmendoza@workstation:~$ sudo apt update
   Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
   Hit:2 http://us.archive.ubuntu.com/ubuntu jammy InRelease
   Get:3 http://us.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
   Get:4 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [319 kB]
   Get:5 http://us.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
   Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [761 kB]
   Get:7 http://us.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [484 kB]
   Get:8 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [164 kB]
   Get:9 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43.0 kB]
   Get:10 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [11.3 kB]
   Get:11 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [822 kB]
   Get:12 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [970 kB]
   Get:13 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [132 kB]

   Get:14 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [781 kB]

   Get:15 http://security.ubuntu.com/ubuntu jammy-security/universe i386 Packages [557 kB]

   Get:16 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [143 kB]

   Get:17 http://us.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [222 kB]

   64% [17 Translation-en 113 kB/222 kB 51%] [Connecting to security.ubuntu.com (185.125.190.36)]
                                                                               Get:18 http://security.ubuntu.com/ubun
   tu jammy-security/universe amd64 DEP-11 Metadata [40.1 kB]

   Get:19 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [101 kB]

   Get:20 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [16.7 kB]
   ```

   ansible all -m apt -a update_cache=true
   What is the result of the command? Is it successful?
   Initially the command was not successful since there is no inventory created yet in the repository. With that, we are unable to run the ansible command. To fix this, we need to create an inventory first.

   Installing the ansible command

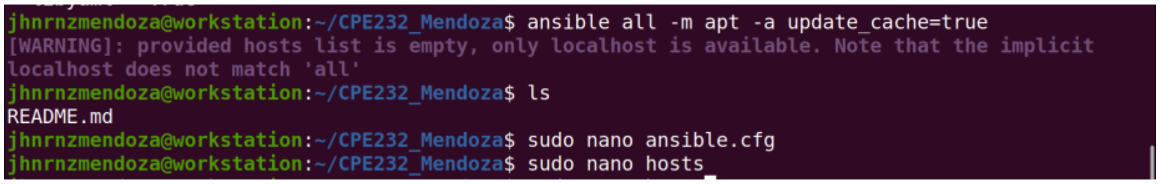
   

   ```
   jhnrnzmendoza@workstation:~/CPE232_MendozaHome$ sudo apt install ansible
   [sudo] password for jhnrnzmendoza:
   Reading package lists... Done
   Building dependency tree... Done
   Reading state information... Done
   ansible is already the newest version (2.10.7+merged+base+2.10.8+dfsg-1).
   The following packages were automatically installed and are no longer required:
     libflashrom1 libftdi1-2 libllvm13
   Use 'sudo apt autoremove' to remove them.
   0 upgraded, 0 newly installed, 0 to remove and 6 not upgraded.
   ```

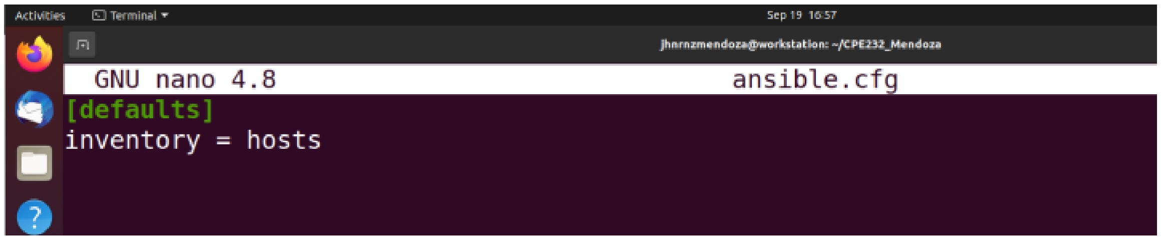


```
jhnrnzmendoza@workstation:~$ ansible all -m apt -a update_cache=true
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not
match 'all'
```

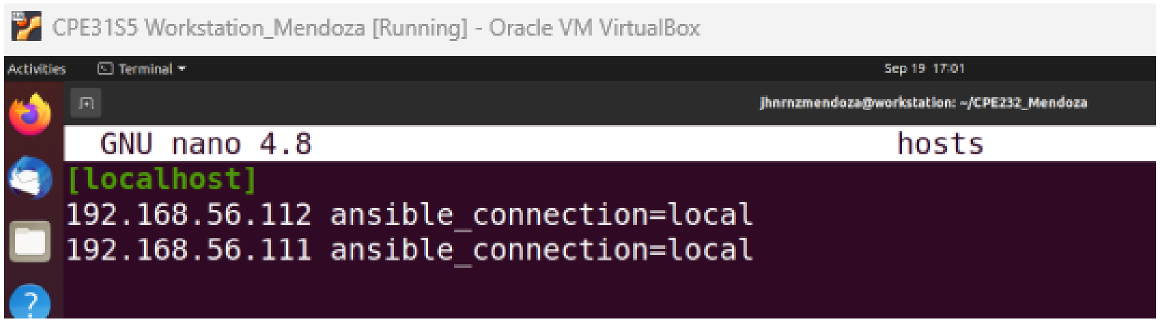
There is no inventory which contains where we would perform ansible. Therefore, we should create.



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible all -m apt -a update_cache=true
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit
localhost does not match 'all'
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ls
README.md
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ sudo nano ansible.cfg
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ sudo nano hosts
```

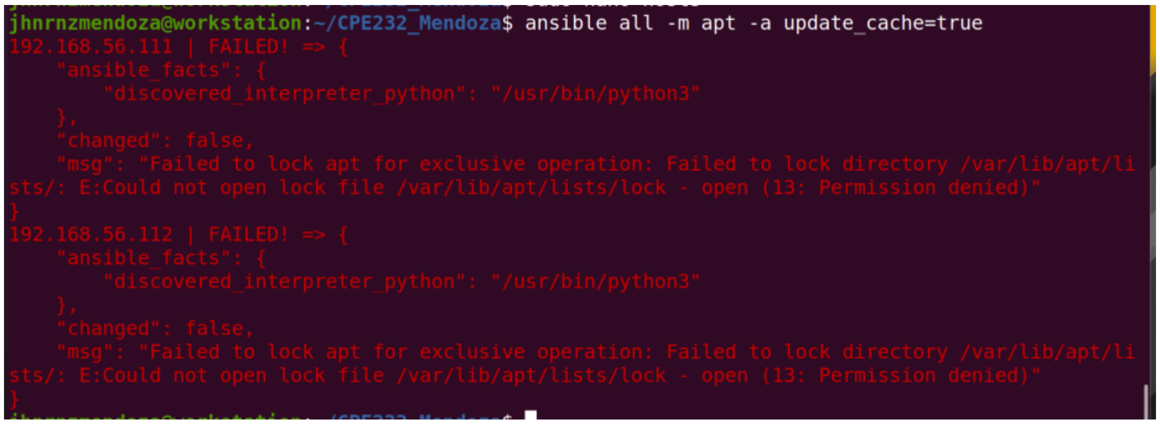


```
  GNU nano 4.8                                ansible.cfg
[defaults]
inventory = hosts
```



CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox

```
  GNU nano 4.8                                ansible.cfg
[localhost]
192.168.56.112 ansible_connection=local
192.168.56.111 ansible_connection=local
```

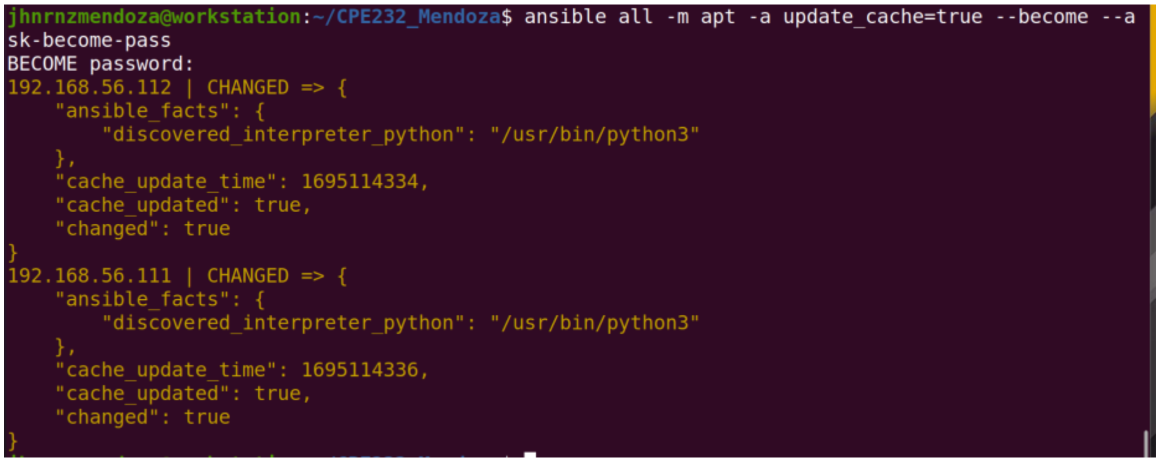


```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible all -m apt -a update_cache=true
192.168.56.111 | FAILED! => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "msg": "Failed to lock apt for exclusive operation: Failed to lock directory /var/lib/apt/li
sts/: E:Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)"
}
192.168.56.112 | 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/li
sts/: E:Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)"
}
```

Observation:

I have used the repository we created in previous activity to store the inventory to be created. The inventory would store the target hosts so that when we run ansible commands, it will cascade to the connected remote servers as well.

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.

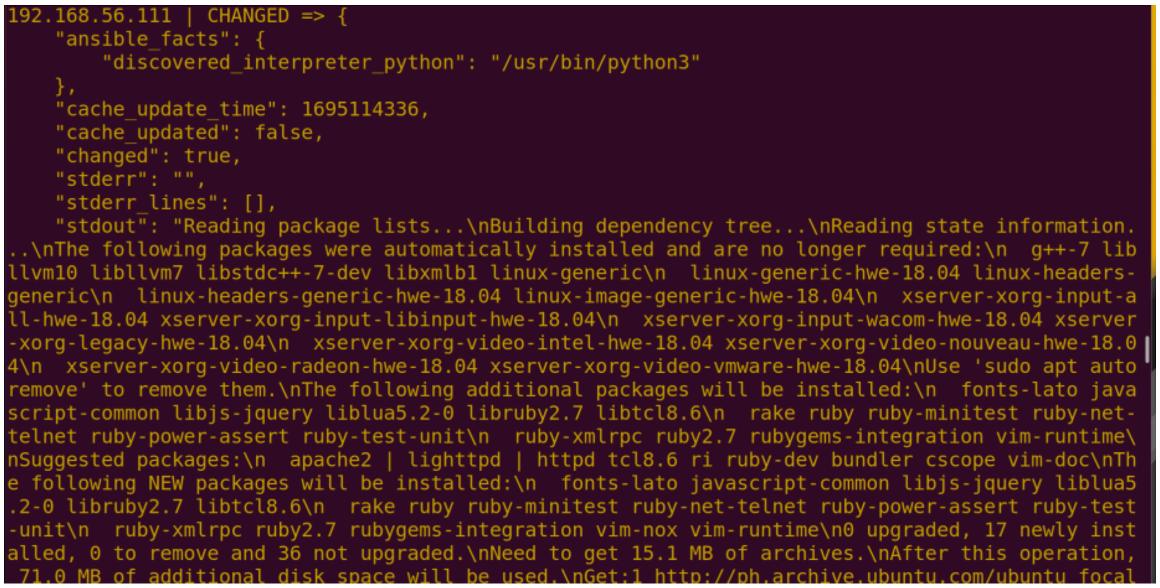


```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible all -m apt -a update_cache=true --become --a
sk-become-pass
BECOME password:
192.168.56.112 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114334,
    "cache_updated": true,
    "changed": true
}
192.168.56.111 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": true,
    "changed": true
}
```

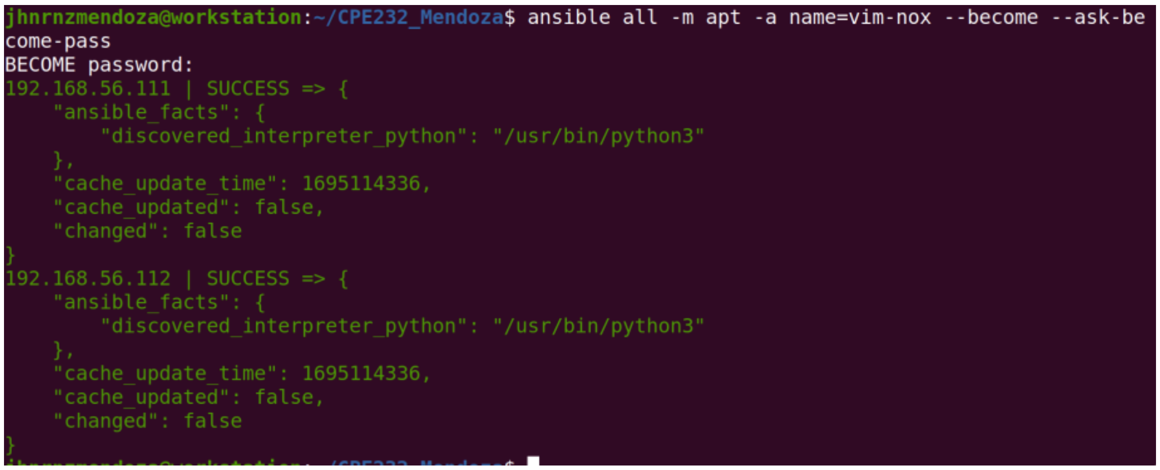
The *update_cache=true* is the same thing as running *sudo apt update*. The --become command elevates the privileges and the *--ask-become-pass* asks for the password. For now, even if we only changed the packaged index, we were able to change something on the remote server.

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

2. Let's try to install VIM, which is an almost compatible version of the UNIX editor Vi. To do this, we will just change 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.



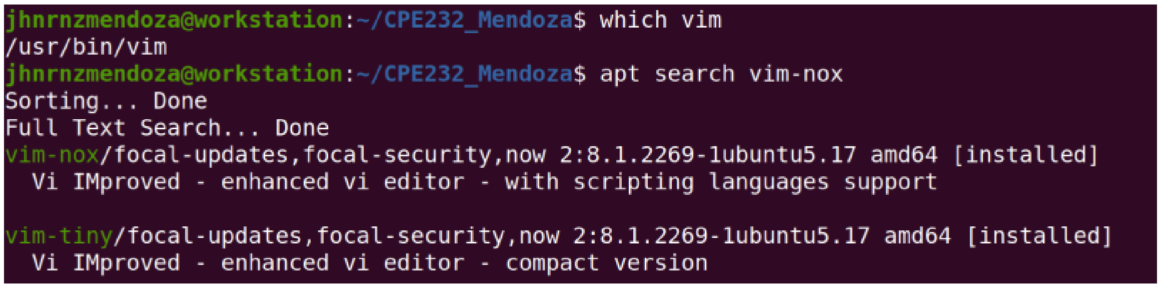
```
192.168.56.111 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading state information.
..\nThe following packages were automatically installed and are no longer required:\n  g++-7 lib
llvm10 libllvm7 libstdc++-7-dev libxmlb1 linux-generic\n  linux-generic-hwe-18.04 linux-headers-
generic\n  linux-headers-generic-hwe-18.04 linux-image-generic-hwe-18.04\n  xserver-xorg-input-a
ll-hwe-18.04 xserver-xorg-input-libinput-hwe-18.04\n  xserver-xorg-input-wacom-hwe-18.04 xserver
-xorg-legacy-hwe-18.04\n  xserver-xorg-video-intel-hwe-18.04 xserver-xorg-video-nouveau-hwe-18.0
4\n  xserver-xorg-video-radeon-hwe-18.04 xserver-xorg-video-vmware-hwe-18.04\nUse 'sudo apt auto
remove' to remove them.\nThe following additional packages will be installed:\n  fonts-lato java
script-common libjs-jquery liblua5.2-0 libruby2.7 libtcl8.6\n  rake ruby ruby-minitest ruby-net-
telnet ruby-power-assert ruby-test-unit\n  ruby-xmlrpc ruby2.7 rubygems-integration vim-runtime\
nSuggested packages:\n  apache2 | lighttpd | httpd tcl8.6 ri ruby-dev bundler cscope vim-doc\nTh
e following NEW packages will be installed:\n  fonts-lato javascript-common libjs-jquery liblua5
.2-0 libruby2.7 libtcl8.6\n  rake ruby ruby-minitest ruby-net-telnet ruby-power-assert ruby-test
-unit\n  ruby-xmlrpc ruby2.7 rubygems-integration vim-nox vim-runtime\n0 upgraded, 17 newly inst
alled, 0 to remove and 36 not upgraded.\nNeed to get 15.1 MB of archives.\nAfter this operation,
 71.0 MB of additional disk space will be used.\nGet:1 http://ph.archive.ubuntu.com/ubuntu focal
```



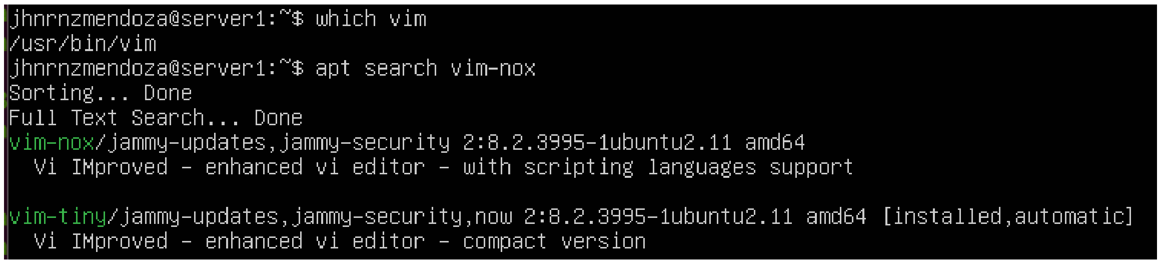
```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible all -m apt -a name=vim-nox --become --ask-be
come-pass
BECOME password:
192.168.56.111 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": false,
    "changed": false
}
192.168.56.112 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": false,
    "changed": false
}
```

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?

Server 1

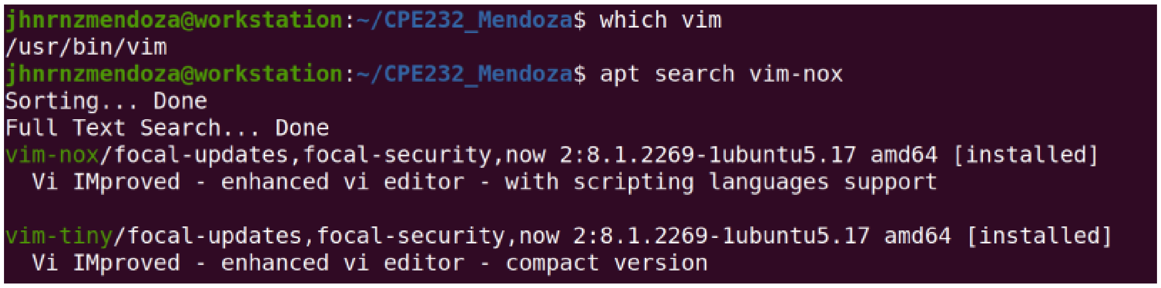


```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ which vim
/usr/bin/vim
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/focal-updates,focal-security,now 2:8.1.2269-1ubuntu5.17 amd64 [installed]
  Vi IMproved - enhanced vi editor - with scripting languages support

vim-tiny/focal-updates,focal-security,now 2:8.1.2269-1ubuntu5.17 amd64 [installed]
  Vi IMproved - enhanced vi editor - compact version
```

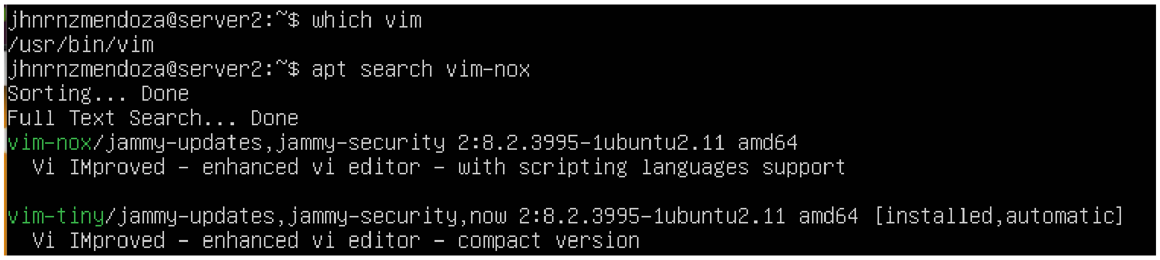


```
jhnrnzmendoza@server1:~$ which vim
/usr/bin/vim
jhnrnzmendoza@server1:~$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/jammy-updates,jammy-security 2:8.2.3995-1ubuntu2.11 amd64
  Vi IMproved - enhanced vi editor - with scripting languages support

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

Server 2



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ which vim
/usr/bin/vim
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/focal-updates,focal-security,now 2:8.1.2269-1ubuntu5.17 amd64 [installed]
  Vi IMproved - enhanced vi editor - with scripting languages support

vim-tiny/focal-updates,focal-security,now 2:8.1.2269-1ubuntu5.17 amd64 [installed]
  Vi IMproved - enhanced vi editor - compact version
```



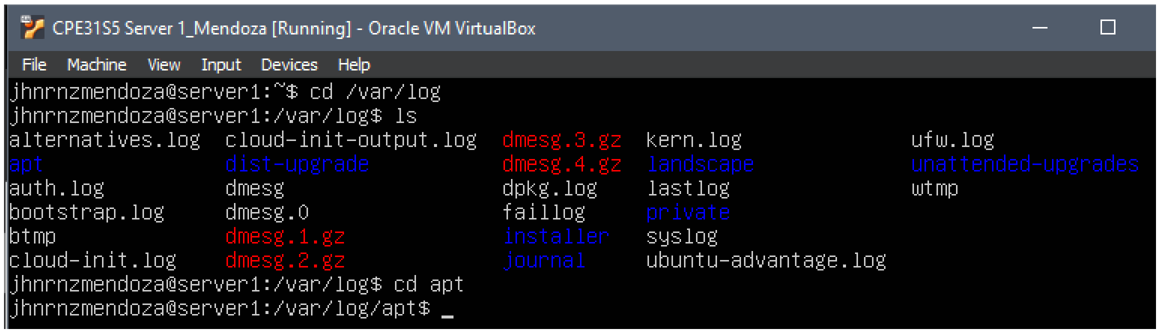
```
jhnrnzmendoza@server2:~$ which vim
/usr/bin/vim
jhnrnzmendoza@server2:~$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/jammy-updates,jammy-security 2:8.2.3995-1ubuntu2.11 amd64
  Vi IMproved - enhanced vi editor - with scripting languages support

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

Observation:

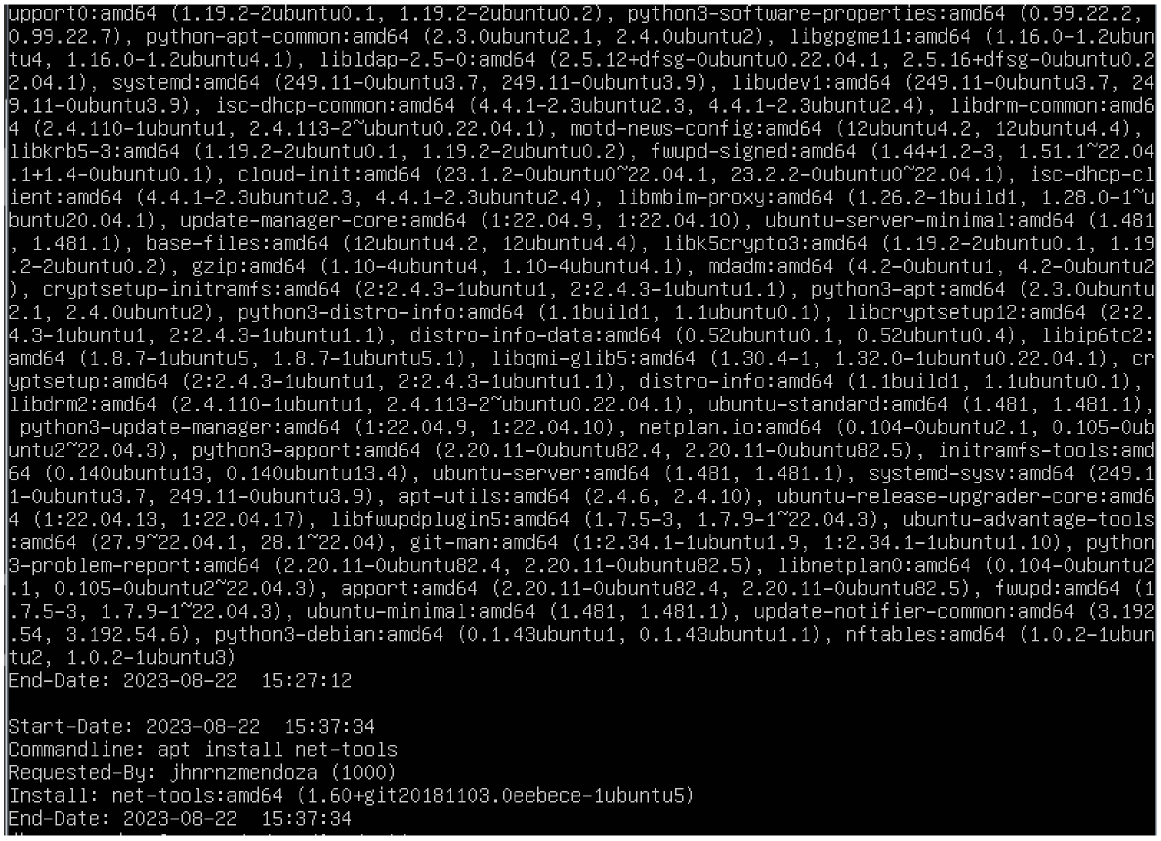
After running the previous step, we have observed that the servers 1 and 2 also had their vim-nox installed.

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.

Server 1

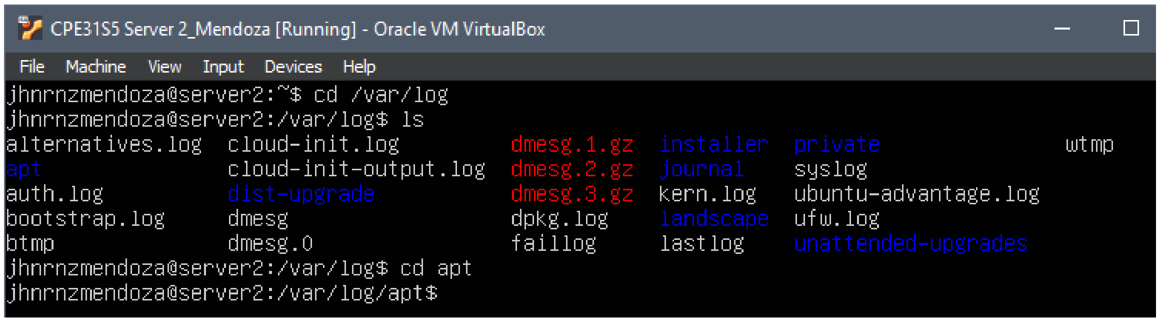

CPE31S5 Server 1_Mendoza [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

```
jhnrnzmendoza@server1:~$ cd /var/log
jhnrnzmendoza@server1:/var/log$ ls
alternatives.log  cloud-init-output.log  dmesg.3.gz  kern.log              ufw.log
apt               dist-upgrade           dmesg.4.gz  landscape             unattended-upgrades
auth.log          dmesg                  dpkg.log    lastlog               wtmp
bootstrap.log     dmesg.0                faillog     private
btmp              dmesg.1.gz             installer   syslog
cloud-init.log    dmesg.2.gz             journal     ubuntu-advantage.log
jhnrnzmendoza@server1:/var/log$ cd apt
jhnrnzmendoza@server1:/var/log/apt$ _
```

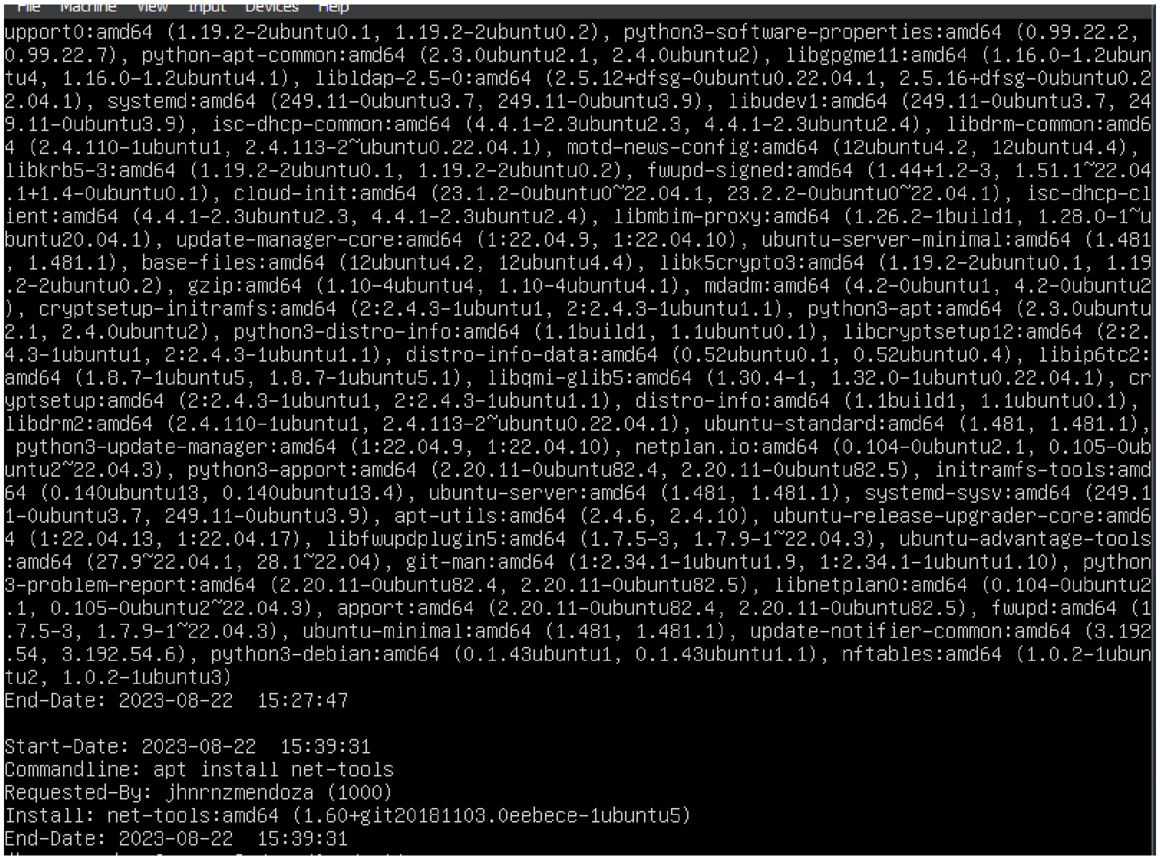


```
upport0:amd64 (1.19.2-2ubuntu0.1, 1.19.2-2ubuntu0.2), python3-software-properties:amd64 (0.99.22.2,
0.99.22.7), python-apt-common:amd64 (2.3.0ubuntu2.1, 2.4.0ubuntu2), libgpgme11:amd64 (1.16.0-1.2ubun
tu4, 1.16.0-1.2ubuntu4.1), libldap-2.5-0:amd64 (2.5.12+dfsg-0ubuntu0.22.04.1, 2.5.16+dfsg-0ubuntu0.2
2.04.1), systemd:amd64 (249.11-0ubuntu3.7, 249.11-0ubuntu3.9), libudev1:amd64 (249.11-0ubuntu3.7, 24
9.11-0ubuntu3.9), isc-dhcp-common:amd64 (4.4.1-2.3ubuntu2.3, 4.4.1-2.3ubuntu2.4), libdrm-common:amd6
4 (2.4.110-1ubuntu1, 2.4.113-2~ubuntu0.22.04.1), motd-news-config:amd64 (12ubuntu4.2, 12ubuntu4.4),
libkrb5-3:amd64 (1.19.2-2ubuntu0.1, 1.19.2-2ubuntu0.2), fwupd-signed:amd64 (1.44+1.2-3, 1.51.1~22.04
.1+1.4-0ubuntu0.1), cloud-init:amd64 (23.1.2-0ubuntu0~22.04.1, 23.2.2-0ubuntu0~22.04.1), isc-dhcp-cl
ient:amd64 (4.4.1-2.3ubuntu2.3, 4.4.1-2.3ubuntu2.4), libmbim-proxy:amd64 (1.26.2-1build1, 1.28.0-1~u
buntu20.04.1), update-manager-core:amd64 (1:22.04.9, 1:22.04.10), ubuntu-server-minimal:amd64 (1.481
, 1.481.1), base-files:amd64 (12ubuntu4.2, 12ubuntu4.4), libk5crypto3:amd64 (1.19.2-2ubuntu0.1, 1.19
.2-2ubuntu0.2), gzip:amd64 (1.10-4ubuntu4, 1.10-4ubuntu4.1), mdadm:amd64 (4.2-0ubuntu1, 4.2-0ubuntu2
), cryptsetup-initramfs:amd64 (2:2.4.3-1ubuntu1, 2:2.4.3-1ubuntu1.1), python3-apt:amd64 (2.3.0ubuntu
2.1, 2.4.0ubuntu2), python3-distro-info:amd64 (1.1build1, 1.1ubuntu0.1), libcryptsetup12:amd64 (2:2.
4.3-1ubuntu1, 2:2.4.3-1ubuntu1.1), distro-info-data:amd64 (0.52ubuntu0.1, 0.52ubuntu0.4), libip6tc2:
amd64 (1.8.7-1ubuntu5, 1.8.7-1ubuntu5.1), libqmi-glib5:amd64 (1.30.4-1, 1.32.0-1ubuntu0.22.04.1), cr
yptsetup:amd64 (2:2.4.3-1ubuntu1, 2:2.4.3-1ubuntu1.1), distro-info:amd64 (1.1build1, 1.1ubuntu0.1),
libdrm2:amd64 (2.4.110-1ubuntu1, 2.4.113-2~ubuntu0.22.04.1), ubuntu-standard:amd64 (1.481, 1.481.1),
 python3-update-manager:amd64 (1:22.04.9, 1:22.04.10), netplan.io:amd64 (0.104-0ubuntu2.1, 0.105-0ub
untu2~22.04.3), python3-apport:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5), initramfs-tools:amd
64 (0.140ubuntu13, 0.140ubuntu13.4), ubuntu-server:amd64 (1.481, 1.481.1), systemd-sysv:amd64 (249.1
1-0ubuntu3.7, 249.11-0ubuntu3.9), apt-utils:amd64 (2.4.6, 2.4.10), ubuntu-release-upgrader-core:amd6
4 (1:22.04.13, 1:22.04.17), libfwupdplugin5:amd64 (1.7.5-3, 1.7.9-1~22.04.3), ubuntu-advantage-tools
:amd64 (27.9~22.04.1, 28.1~22.04), git-man:amd64 (1:2.34.1-1ubuntu1.9, 1:2.34.1-1ubuntu1.10), python
3-problem-report:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5), libnetplan0:amd64 (0.104-0ubuntu2
.1, 0.105-0ubuntu2~22.04.3), apport:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5), fwupd:amd64 (1
.7.5-3, 1.7.9-1~22.04.3), ubuntu-minimal:amd64 (1.481, 1.481.1), update-notifier-common:amd64 (3.192
.54, 3.192.54.6), python3-debian:amd64 (0.1.43ubuntu1, 0.1.43ubuntu1.1), nftables:amd64 (1.0.2-1ubun
tu2, 1.0.2-1ubuntu3)
End-Date: 2023-08-22  15:27:12

Start-Date: 2023-08-22  15:37:34
Commandline: apt install net-tools
Requested-By: jhnrnzmendoza (1000)
Install: net-tools:amd64 (1.60+git20181103.0eebece-1ubuntu5)
End-Date: 2023-08-22  15:37:34
```

Server 2

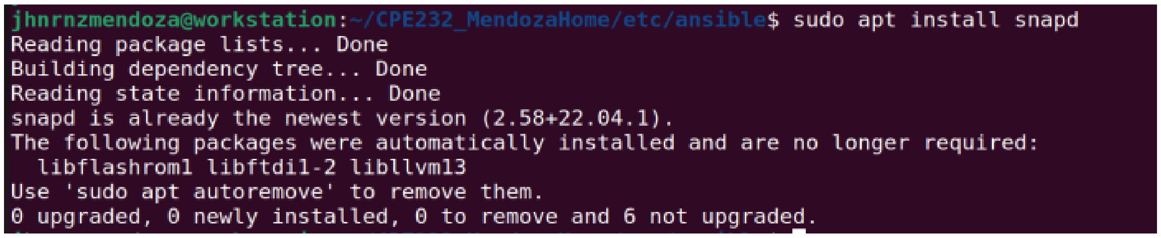

CPE31S5 Server 2_Mendoza [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

```
jhnrnzmendoza@server2:~$ cd /var/log
jhnrnzmendoza@server2:/var/log$ ls
alternatives.log  cloud-init.log         dmesg.1.gz  installer  private                  wtmp
apt               cloud-init-output.log  dmesg.2.gz  journal    syslog
auth.log          dist-upgrade           dmesg.3.gz  kern.log   ubuntu-advantage.log
bootstrap.log     dmesg                  dpkg.log    landscape  ufw.log
btmp              dmesg.0                faillog     lastlog    unattended-upgrades
jhnrnzmendoza@server2:/var/log$ cd apt
jhnrnzmendoza@server2:/var/log/apt$
```



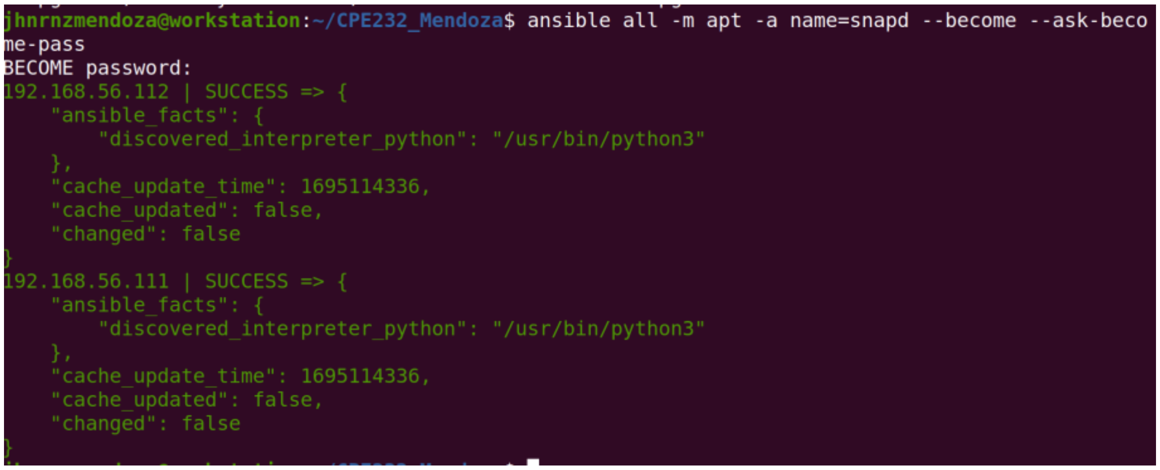
```
upport0:amd64 (1.19.2-2ubuntu0.1, 1.19.2-2ubuntu0.2), python3-software-properties:amd64 (0.99.22.2,
0.99.22.7), python-apt-common:amd64 (2.3.0ubuntu2.1, 2.4.0ubuntu2), libgpgme11:amd64 (1.16.0-1.2ubun
tu4, 1.16.0-1.2ubuntu4.1), libldap-2.5-0:amd64 (2.5.12+dfsg-0ubuntu0.22.04.1, 2.5.16+dfsg-0ubuntu0.2
2.04.1), systemd:amd64 (249.11-0ubuntu3.7, 249.11-0ubuntu3.9), libudev1:amd64 (249.11-0ubuntu3.7, 24
9.11-0ubuntu3.9), isc-dhcp-common:amd64 (4.4.1-2.3ubuntu2.3, 4.4.1-2.3ubuntu2.4), libdrm-common:amd6
4 (2.4.110-1ubuntu1, 2.4.113-2~ubuntu0.22.04.1), motd-news-config:amd64 (12ubuntu4.2, 12ubuntu4.4),
libkrb5-3:amd64 (1.19.2-2ubuntu0.1, 1.19.2-2ubuntu0.2), fwupd-signed:amd64 (1.44+1.2-3, 1.51.1~22.04
.1+1.4-0ubuntu0.1), cloud-init:amd64 (23.1.2-0ubuntu0~22.04.1, 23.2.2-0ubuntu0~22.04.1), isc-dhcp-cl
ient:amd64 (4.4.1-2.3ubuntu2.3, 4.4.1-2.3ubuntu2.4), libmbim-proxy:amd64 (1.26.2-1build1, 1.28.0-1~u
buntu20.04.1), update-manager-core:amd64 (1:22.04.9, 1:22.04.10), ubuntu-server-minimal:amd64 (1.481
, 1.481.1), base-files:amd64 (12ubuntu4.2, 12ubuntu4.4), libk5crypto3:amd64 (1.19.2-2ubuntu0.1, 1.19
.2-2ubuntu0.2), gzip:amd64 (1.10-4ubuntu4, 1.10-4ubuntu4.1), mdadm:amd64 (4.2-0ubuntu1, 4.2-0ubuntu2
), cryptsetup-initramfs:amd64 (2:2.4.3-1ubuntu1, 2:2.4.3-1ubuntu1.1), python3-apt:amd64 (2.3.0ubuntu
2.1, 2.4.0ubuntu2), python3-distro-info:amd64 (1.1build1, 1.1ubuntu0.1), libcryptsetup12:amd64 (2:2.
4.3-1ubuntu1, 2:2.4.3-1ubuntu1.1), distro-info-data:amd64 (0.52ubuntu0.1, 0.52ubuntu0.4), libip6tc2:
amd64 (1.8.7-1ubuntu5, 1.8.7-1ubuntu5.1), libqmi-glib5:amd64 (1.30.4-1, 1.32.0-1ubuntu0.22.04.1), cr
yptsetup:amd64 (2:2.4.3-1ubuntu1, 2:2.4.3-1ubuntu1.1), distro-info:amd64 (1.1build1, 1.1ubuntu0.1),
libdrm2:amd64 (2.4.110-1ubuntu1, 2.4.113-2~ubuntu0.22.04.1), ubuntu-standard:amd64 (1.481, 1.481.1),
 python3-update-manager:amd64 (1:22.04.9, 1:22.04.10), netplan.io:amd64 (0.104-0ubuntu2.1, 0.105-0ub
untu2~22.04.3), python3-apport:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5), initramfs-tools:amd
64 (0.140ubuntu13, 0.140ubuntu13.4), ubuntu-server:amd64 (1.481, 1.481.1), systemd-sysv:amd64 (249.1
1-0ubuntu3.7, 249.11-0ubuntu3.9), apt-utils:amd64 (2.4.6, 2.4.10), ubuntu-release-upgrader-core:amd6
4 (1:22.04.13, 1:22.04.17), libfwupdplugin5:amd64 (1.7.5-3, 1.7.9-1~22.04.3), ubuntu-advantage-tools
:amd64 (27.9~22.04.1, 28.1~22.04), git-man:amd64 (1:2.34.1-1ubuntu1.9, 1:2.34.1-1ubuntu1.10), python
3-problem-report:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5), libnetplan0:amd64 (0.104-0ubuntu2
.1, 0.105-0ubuntu2~22.04.3), apport:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5), fwupd:amd64 (1
.7.5-3, 1.7.9-1~22.04.3), ubuntu-minimal:amd64 (1.481, 1.481.1), update-notifier-common:amd64 (3.192
.54, 3.192.54.6), python3-debian:amd64 (0.1.43ubuntu1, 0.1.43ubuntu1.1), nftables:amd64 (1.0.2-1ubun
tu2, 1.0.2-1ubuntu3)
End-Date: 2023-08-22  15:27:47

Start-Date: 2023-08-22  15:39:31
Commandline: apt install net-tools
Requested-By: jhnrnzmendoza (1000)
Install: net-tools:amd64 (1.60+git20181103.0eebece-1ubuntu5)
End-Date: 2023-08-22  15:39:31
```

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



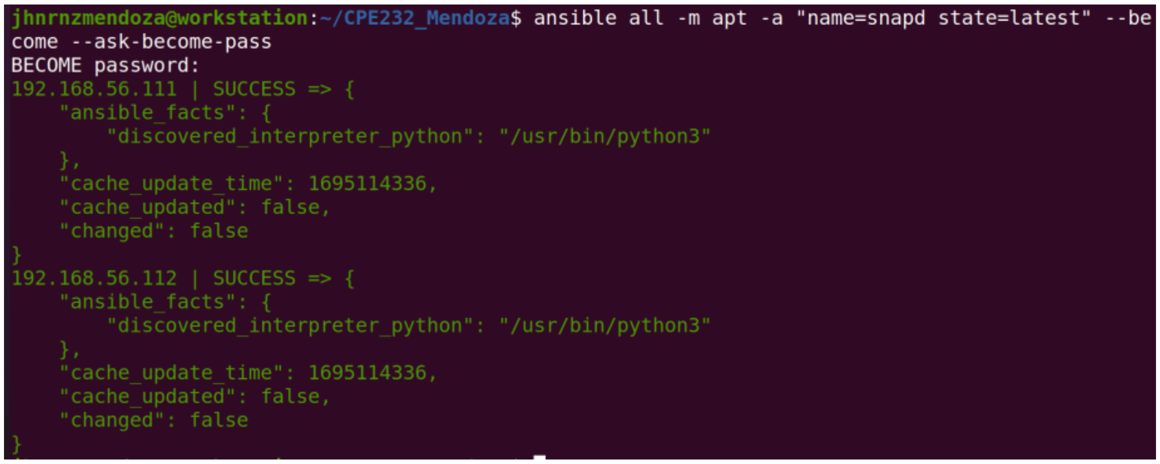
```
jhnrnzmendoza@workstation:~/CPE232_MendozaHome/etc/ansible$ sudo apt install snapd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
snapd is already the newest version (2.58+22.04.1).
The following packages were automatically installed and are no longer required:
  libflashrom1 libftdi1-2 libllvm13
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 6 not upgraded.
```

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?



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible all -m apt -a name=snapd --become --ask-beco
me-pass
BECOME password:
192.168.56.112 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": false,
    "changed": false
}
192.168.56.111 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": false,
    "changed": false
}
```

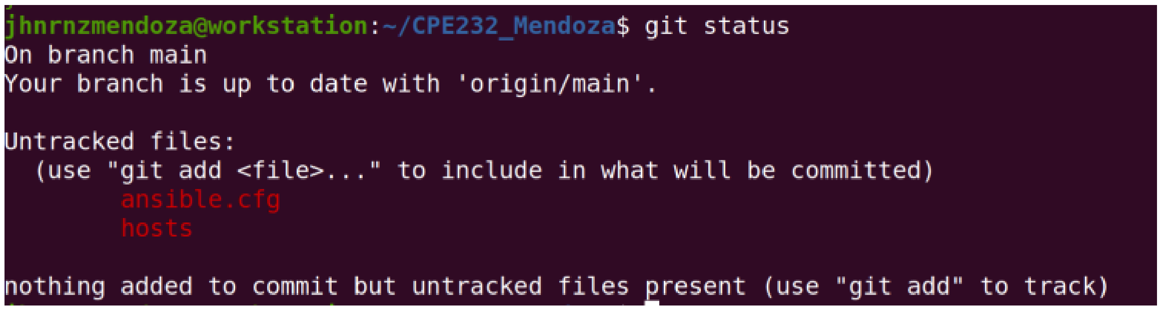
3.2 Now, try to issue this command: *ansible all -m apt -a "name=snapd state=latest" --become --ask-become-pass*
Describe the output of this command. Notice how we added the command *state=latest* and placed them in double quotations.



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible all -m apt -a "name=snapd state=latest" --be
come --ask-become-pass
BECOME password:
192.168.56.111 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": false,
    "changed": false
}
192.168.56.112 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1695114336,
    "cache_updated": false,
    "changed": false
}
```

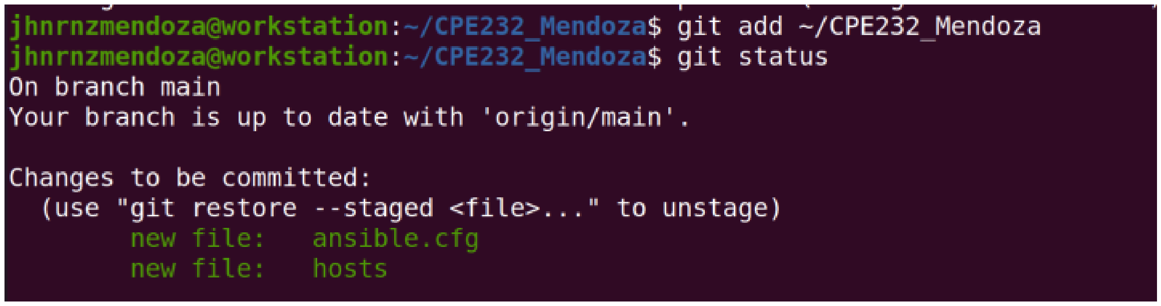
4. At this point, make sure to commit all changes to GitHub.

Step 1: Check the status of the repository.



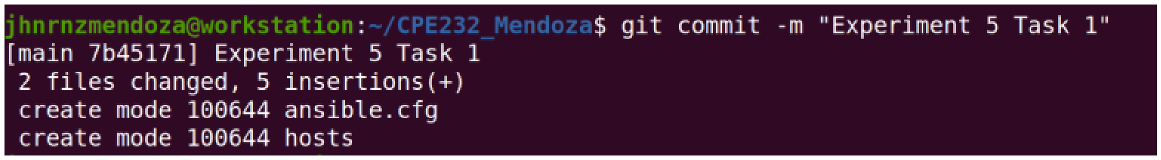
```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ansible.cfg
        hosts

nothing added to commit but untracked files present (use "git add" to track)
```

Step 2: Use the command git add to make changes to the repository. Check the status again if the add is successful.



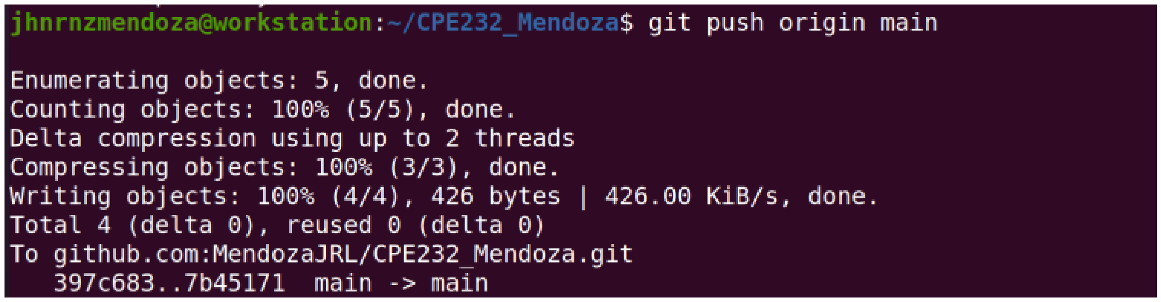
```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git add ~/CPE232_Mendoza
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   ansible.cfg
        new file:   hosts
```

Step 3: Commit the changes to the repository

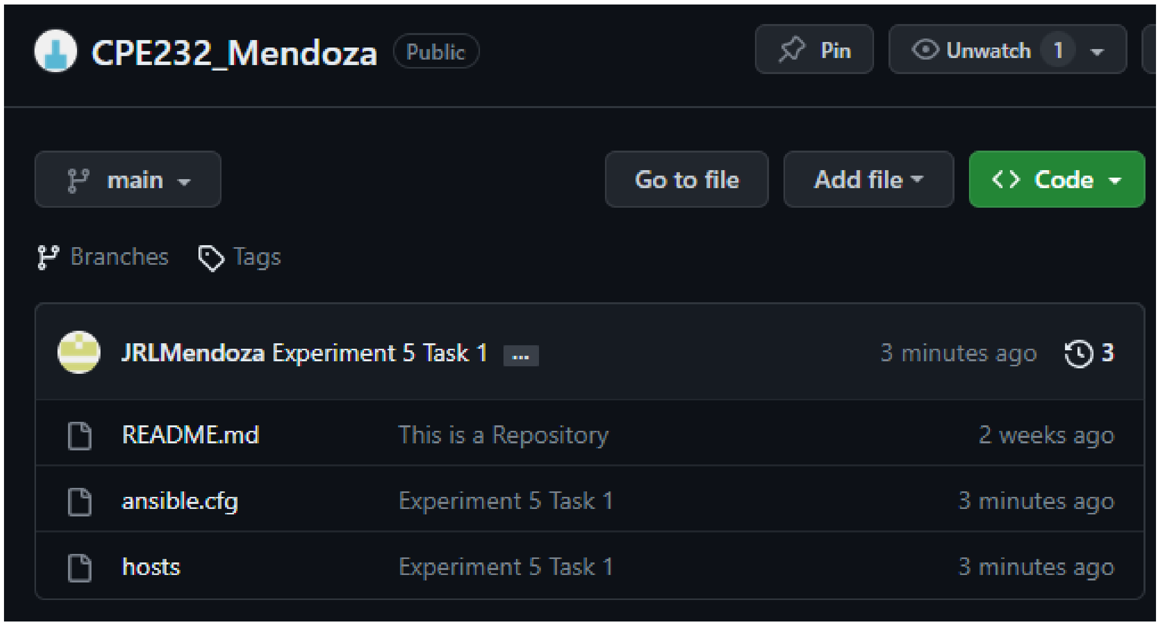


```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git commit -m "Experiment 5 Task 1"
[main 7b45171] Experiment 5 Task 1
 2 files changed, 5 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 hosts
```

Step 4: Push the changes to GitHub. If asked for a private key, input the root password.



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ 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% (4/4), 426 bytes | 426.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To github.com:MendozaJRL/CPE232_Mendoza.git
   397c683..7b45171  main -> main
```


CPE232_Mendoza Public

Pin

Unwatch 1

main

Go to file

Add file

Code

Branches Tags

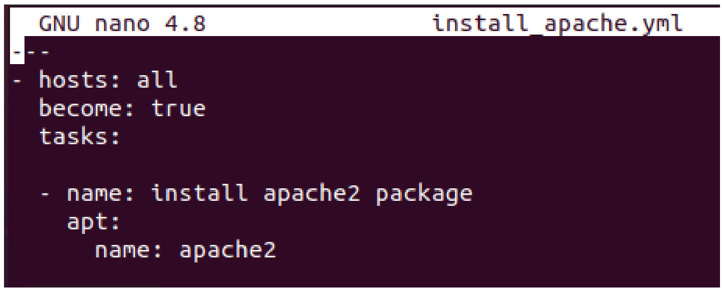
JRLMendoza Experiment 5 Task 1 ...		3 minutes ago 3
README.md	This is a Repository	2 weeks ago
ansible.cfg	Experiment 5 Task 1	3 minutes ago
hosts	Experiment 5 Task 1	3 minutes ago

Observation:
Using the previously used commands on task 1 of experiment 3. We are able to commit the changes on the repository by using the git commands such as: git add, git commit, and git push. We can also observe that the repository has new contents which are the ansible.cfg and hosts file.

Task 2: Writing our First Playbook

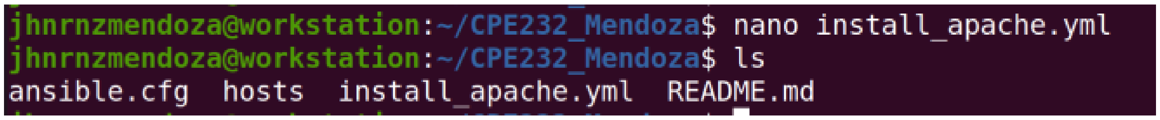
1. With ad hoc commands, we can simplify the administration of remote servers. For example, we can install updates, packages, and applications, etc. However, the real strength of Ansible comes from its playbooks. When we write a playbook, we can define the state that we want our servers to be in and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

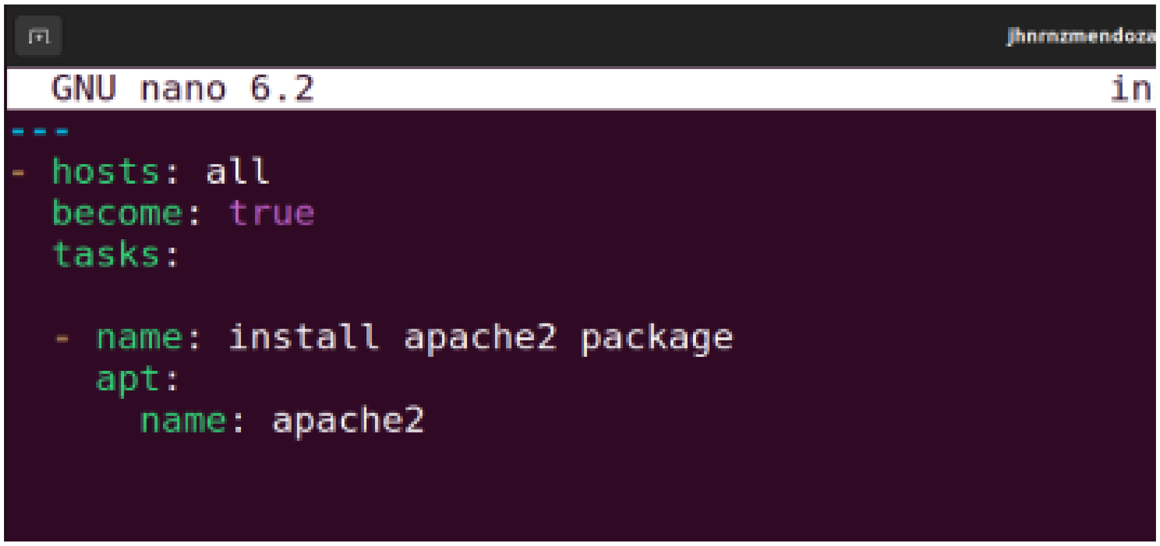


```
  GNU nano 4.8                    install_apache.yml
---
- hosts: all
  become: true
  tasks:

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

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

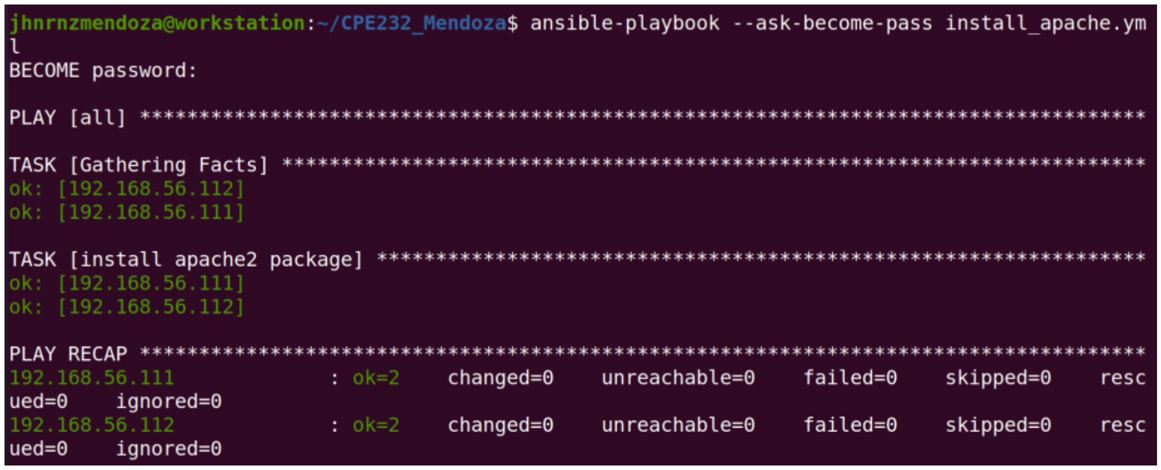


```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ nano install_apache.yml
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ls
ansible.cfg  hosts  install_apache.yml  README.md
```



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

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

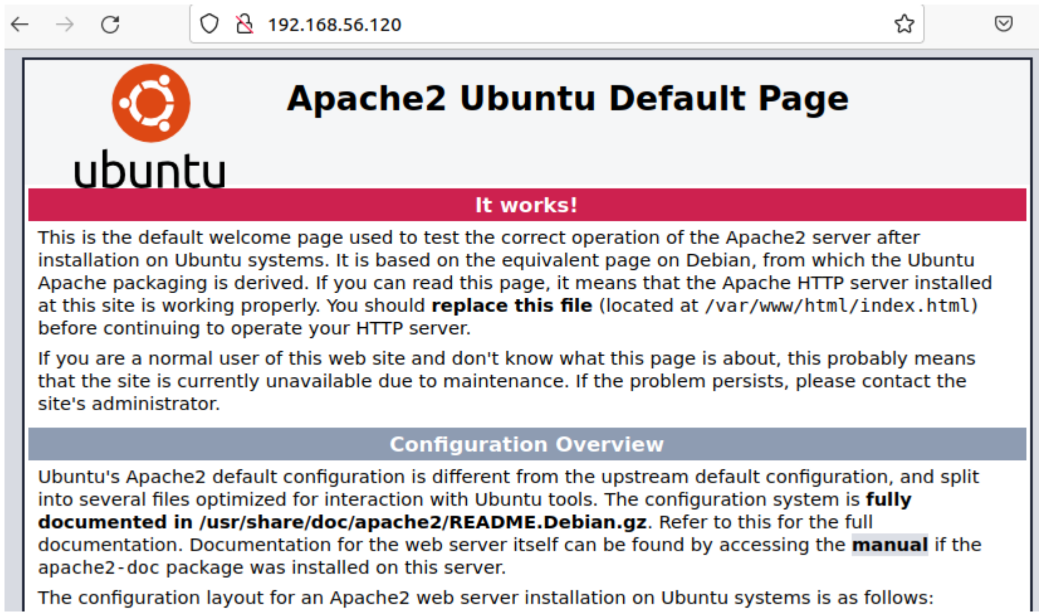
2. Run the yml file using the command: *ansible-playbook --ask-become-pass install_apache.yml.* Describe the result of this command.



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible-playbook --ask-become-pass install_apache.ym
l
BECOME password:

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

TASK [Gathering Facts] *********************************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

TASK [install apache2 package] *************************************************************
ok: [192.168.56.111]
ok: [192.168.56.112]

PLAY RECAP *********************************************************************************
192.168.56.111             : ok=2    changed=0    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
192.168.56.112             : ok=2    changed=0    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
```

By running this command, we can observe that the ip addresses that we have included in the hosts file were able to be installed with the apache2 package.

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.



Apache2 Ubuntu Default Page

It works!

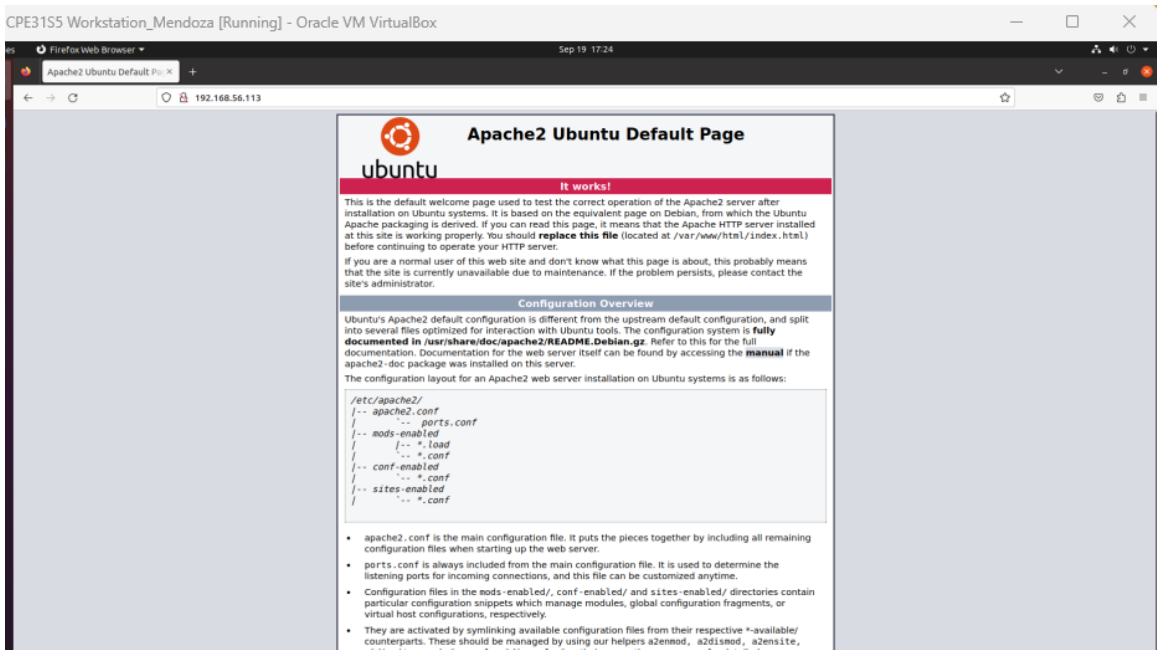
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at 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:


CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox

Sep 19 17:24

192.168.56.113

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.

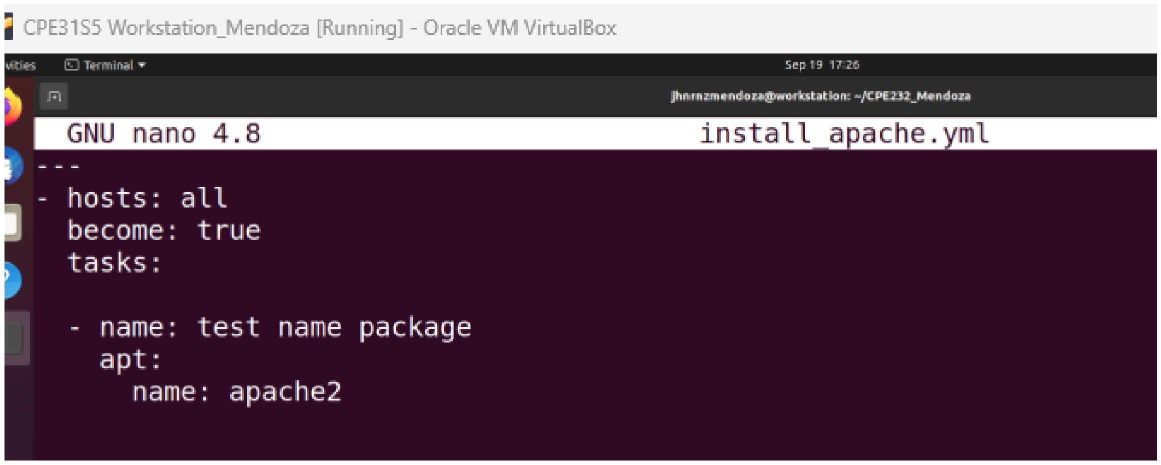
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/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.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`,

Observation:
The servers on our virtual box were installed with the live-server of the ubuntu distribution. With that, we are unable to navigate its local browser since it is only in the command line interface.

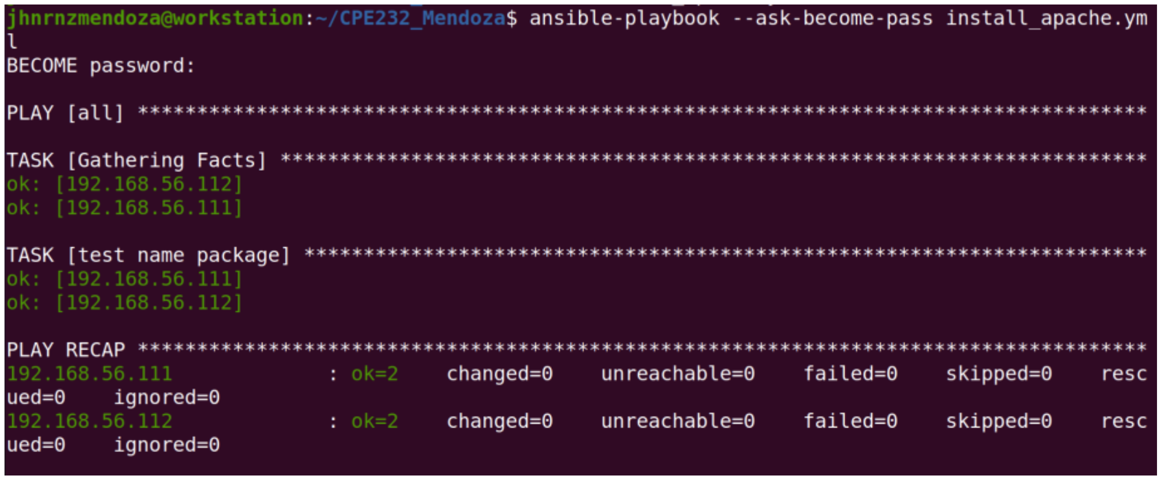
4. Try to edit the *install_apache.yml* and change the name of the package to any name that will not be recognized. What is the output?


CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox

Sep 19 17:26

jhnrnzmendoza@workstation: ~/CPE232_Mendoza

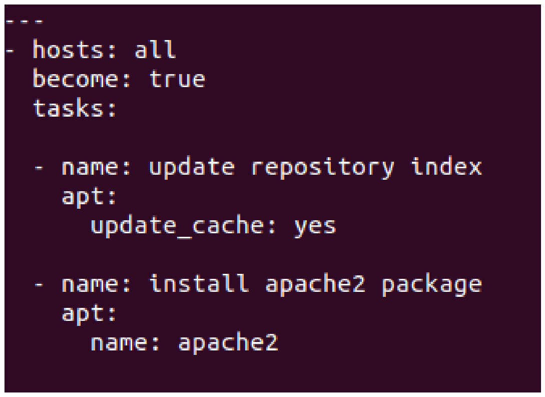
```
  GNU nano 4.8                                         install_apache.yml
---
- hosts: all
  become: true
  tasks:

  - name: test name package
    apt:
      name: apache2
```



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible-playbook --ask-become-pass install_apache.ym
l
BECOME password:

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

TASK [Gathering Facts] *********************************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

TASK [test name package] *******************************************************************
ok: [192.168.56.111]
ok: [192.168.56.112]

PLAY RECAP *********************************************************************************
192.168.56.111             : ok=2    changed=0    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
192.168.56.112             : ok=2    changed=0    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
```

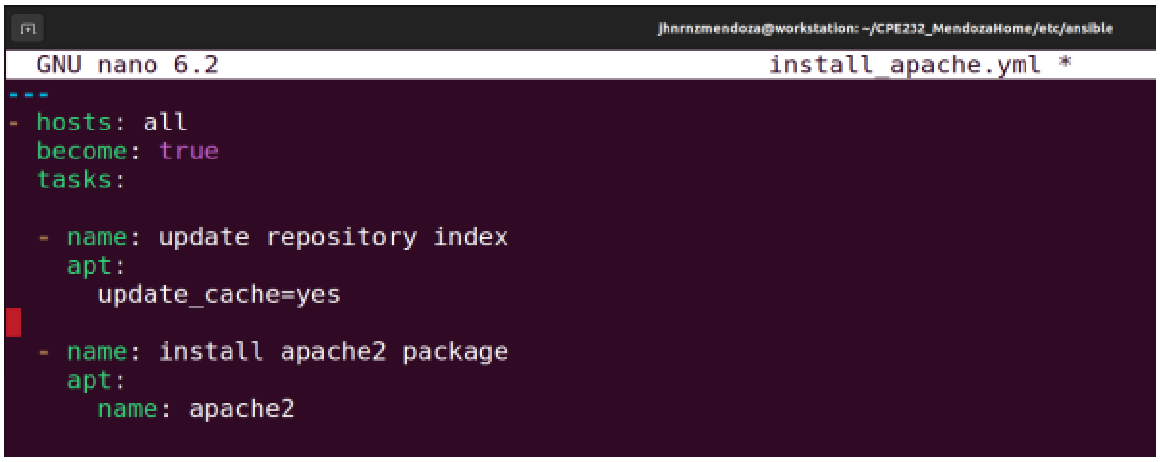
Observation:
Even with a different name of the playbook. The commands were run successfully as the definition of the actual command to be performed on the connected remote hosts were not changed.

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.



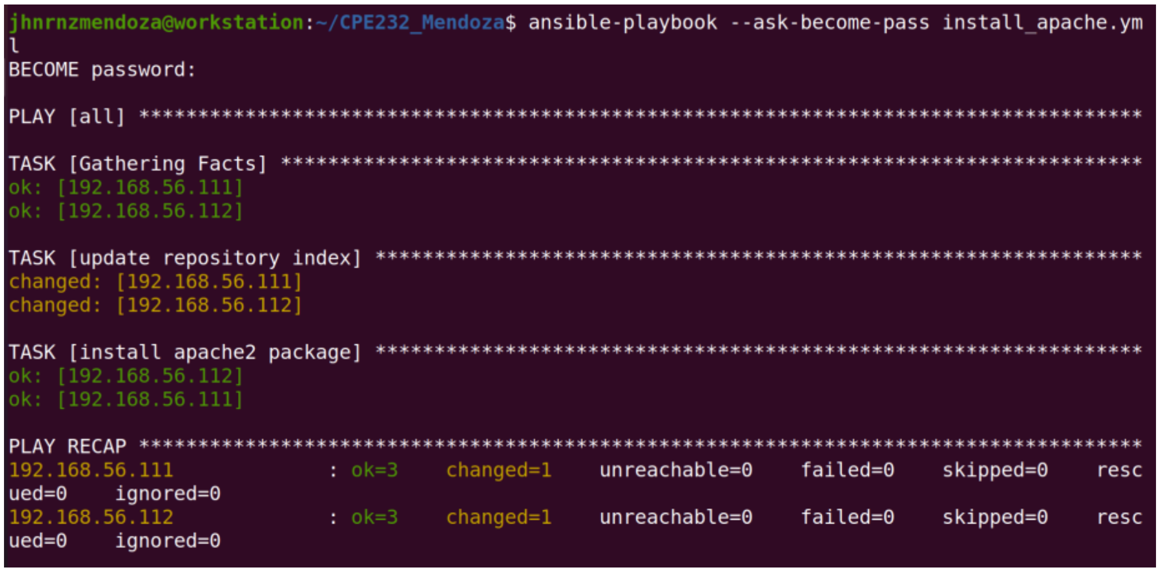
```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes

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

Save the changes to this file and exit.



```
jhnrnzmendoza@workstation: ~/CPE232_MendozaHome/etc/ansible
 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
```

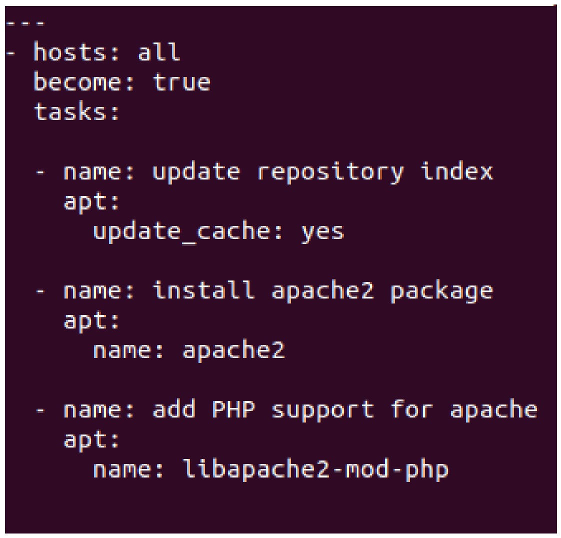
6. Run the playbook and describe the output. Did the new command change anything on the remote servers?



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible-playbook --ask-become-pass install_apache.ym
l
BECOME password:

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

TASK [Gathering Facts] *********************************************************************
ok: [192.168.56.111]
ok: [192.168.56.112]

TASK [update repository index] *************************************************************
changed: [192.168.56.111]
changed: [192.168.56.112]

TASK [install apache2 package] *************************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

PLAY RECAP *********************************************************************************
192.168.56.111             : ok=3    changed=1    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
192.168.56.112             : ok=3    changed=1    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
```

Observation:
By looking at the output of the command, we can observe that both of the ip addresses for server 1 and server 2 were successfully configured. To further check if it is successful, we may try to use the servers and manually check the changes.

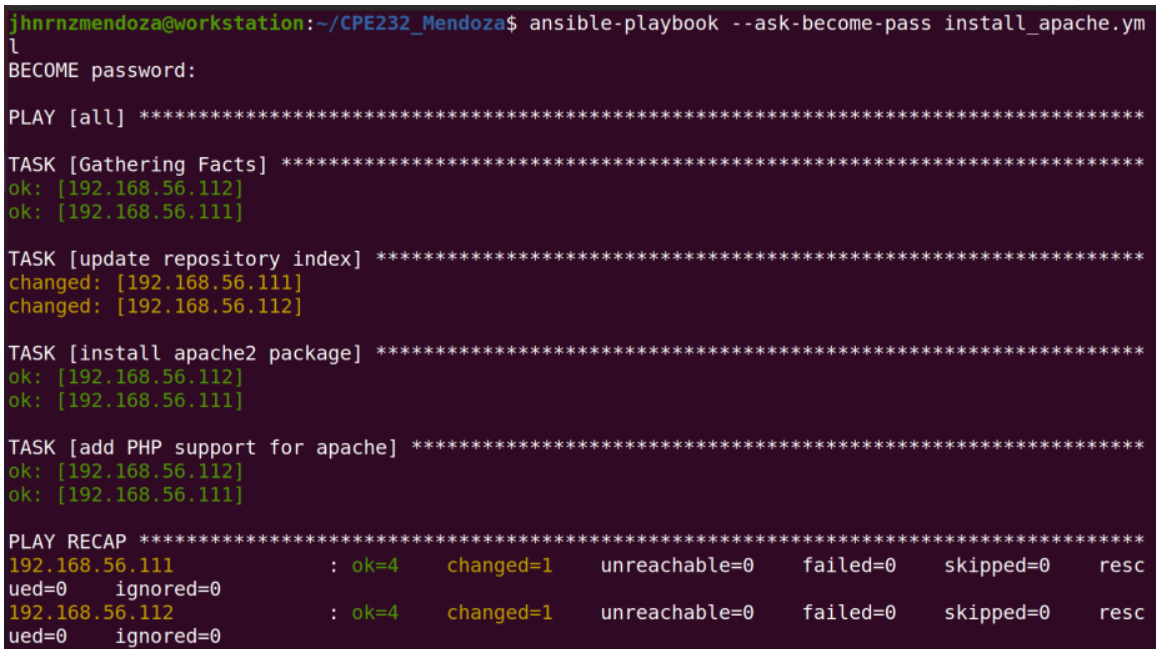
7. Edit again the *install_apache.yml*. This time, we are going to add a PHP support for the apache package we installed earlier.



```
---
- 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.

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



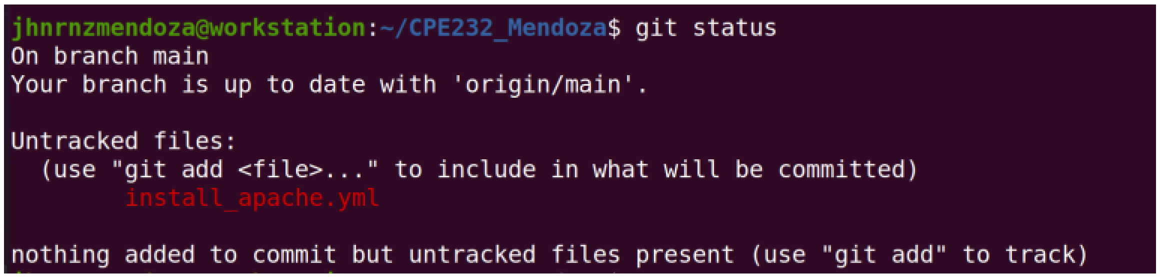
```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ ansible-playbook --ask-become-pass install_apache.ym
l
BECOME password:

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

TASK [Gathering Facts] *********************************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

TASK [update repository index] *************************************************************
changed: [192.168.56.111]
changed: [192.168.56.112]

TASK [install apache2 package] *************************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

TASK [add PHP support for apache] **********************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

PLAY RECAP *********************************************************************************
192.168.56.111             : ok=4    changed=1    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
192.168.56.112             : ok=4    changed=1    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
```

Observation:

Observing the results of the command, we may say that the command was successful and it was able to make configurations to the remote servers.

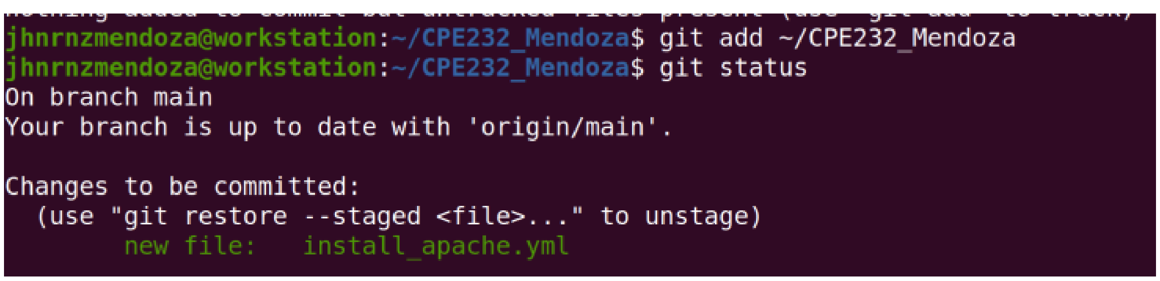
9. Finally, make sure that we are in sync with GitHub. Provide the link of your GitHub repository.

Step 1: Check the status of the repository.



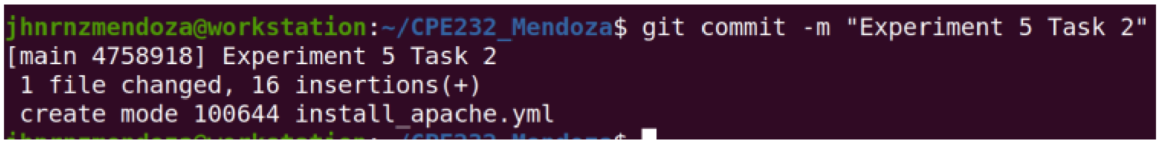
```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        install_apache.yml

nothing added to commit but untracked files present (use "git add" to track)
```

Step 2: Use the command git add to make changes to the repository. Check the status again if the add is successful.



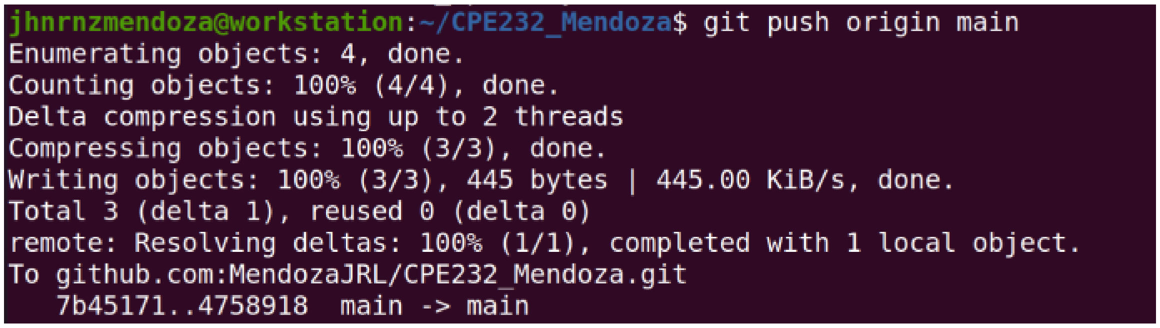
```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git add ~/CPE232_Mendoza
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   install_apache.yml
```

Step 3: Commit the changes to the repository

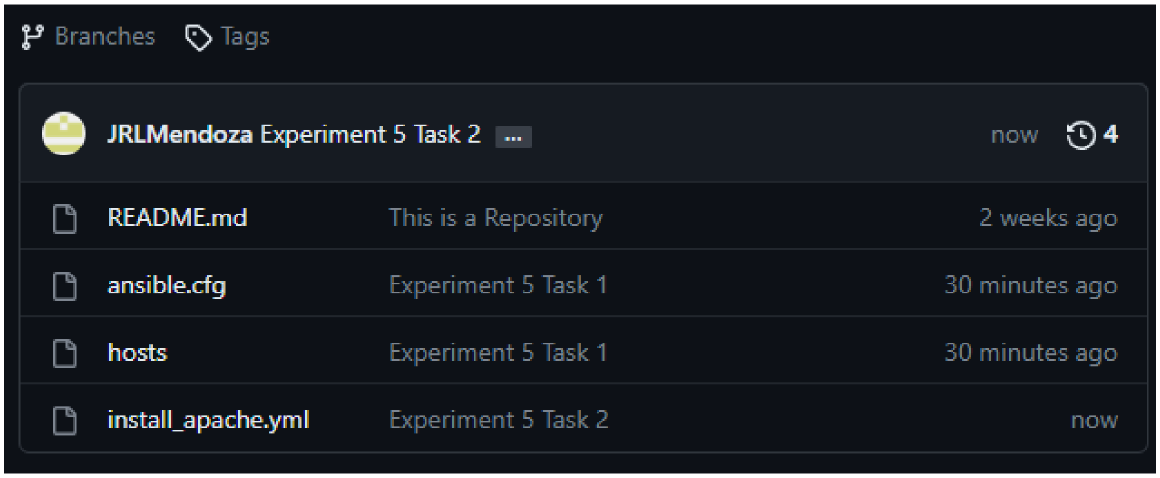


```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git commit -m "Experiment 5 Task 2"
[main 4758918] Experiment 5 Task 2
 1 file changed, 16 insertions(+)
 create mode 100644 install_apache.yml
```

Step 4: Push the changes to GitHub. If asked for a private key, input the root password.



```
jhnrnzmendoza@workstation:~/CPE232_Mendoza$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 445 bytes | 445.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:MendozaJRL/CPE232_Mendoza.git
   7b45171..4758918  main -> main
```



README.md	This is a Repository	2 weeks ago
ansible.cfg	Experiment 5 Task 1	30 minutes ago
hosts	Experiment 5 Task 1	30 minutes ago
install_apache.yml	Experiment 5 Task 2	now

Observation:
Using the previously used commands of experiment 3. We are able to commit the changes on the repository by using the git commands such as: git add, git commit, and git push. We can also observe that the repository has new content which is the playbook created on experiment 4.

Github Link: https://github.com/MendozaJRL/CPE232_Mendoza.git

Note: Pushed Changes from Task 1 and Task 2 is on the same repository

Reflections:

Answer the following:

1. What is the importance of using a playbook?

   The playbook is a file used on ansible in a .yml file extension. A playbook contains a set of commands similar to a script that can run simultaneously when the playbook is executed. Since playbook is used with ansible and git. It is extremely useful in performing automation with multiple hosts or servers. As we have observed with task 2. The students created a playbook that installed apache2, updated apache2, and added PHP support. We only created one playbook file and since we have multiple servers connected to our local host. We are able to observe the playbook do the series of commands on all of the remote servers without manually doing them at a time. Moreover, this shows that using a playbook makes system administrator job efficient and easier.

2. Summarize what we have done on this activity.

   In this laboratory experiment, the students were able to install ansible and create a playbook in order to configure multiple devices at once using one local host. For the first task, the students installed ansible on the git repository done in the previous experiment, together with the appropriate target inventory and hosts. Using this initial configuration, we have installed the snapd on the localhost as well as on the connected remote servers. But this process is quite tedious as you will declare the command every time you want to configure the localhost and connected devices. Which is why, in the second task, the students created a playbook that can store a series of commands so that when the playbook is executed, the commands would run simultaneously and with the addition of ansible, we are able to perform the playbook on multiple devices. Lastly, the students committed the changes of the local repository to the github by performing git commands such as: git add, git commit, and git push origin main. By committing the changes, the students would be able to keep track of their progress in the github web repository.