Name: Villasenor, Hans Rainier A.	**Date Performed: 08/28/25**
Course/Section: CPE 212 - CPE31S2	**Date Submitted: 08/28/25**
Instructor: Engr. Robin Valenzuela	**Semester and SY:**

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.

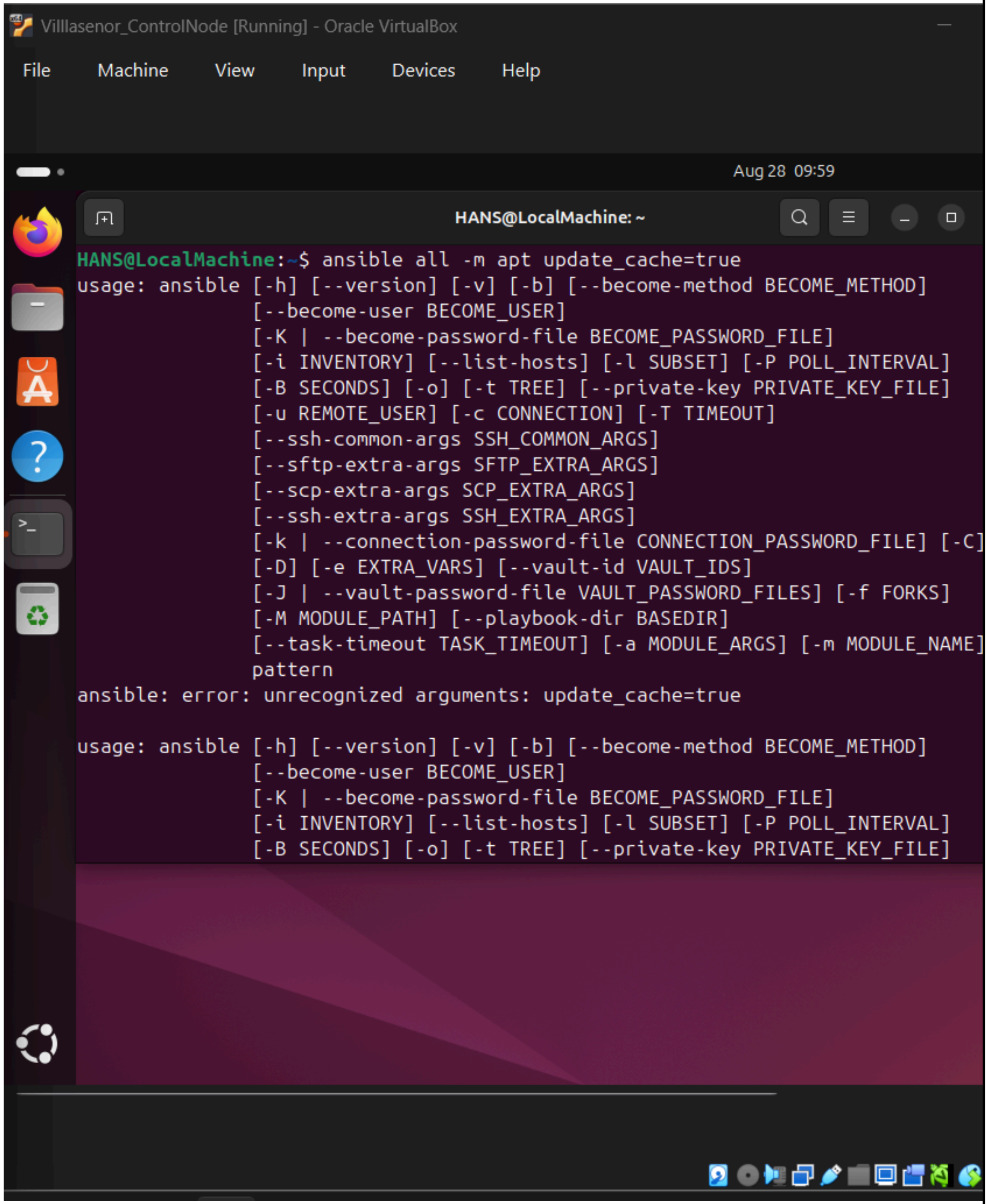
Playbooks record and execute **Ansible**'s configuration, deployment, and orchestration functions. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. If Ansible modules are the tools in your workshop, playbooks are your instruction manuals, 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



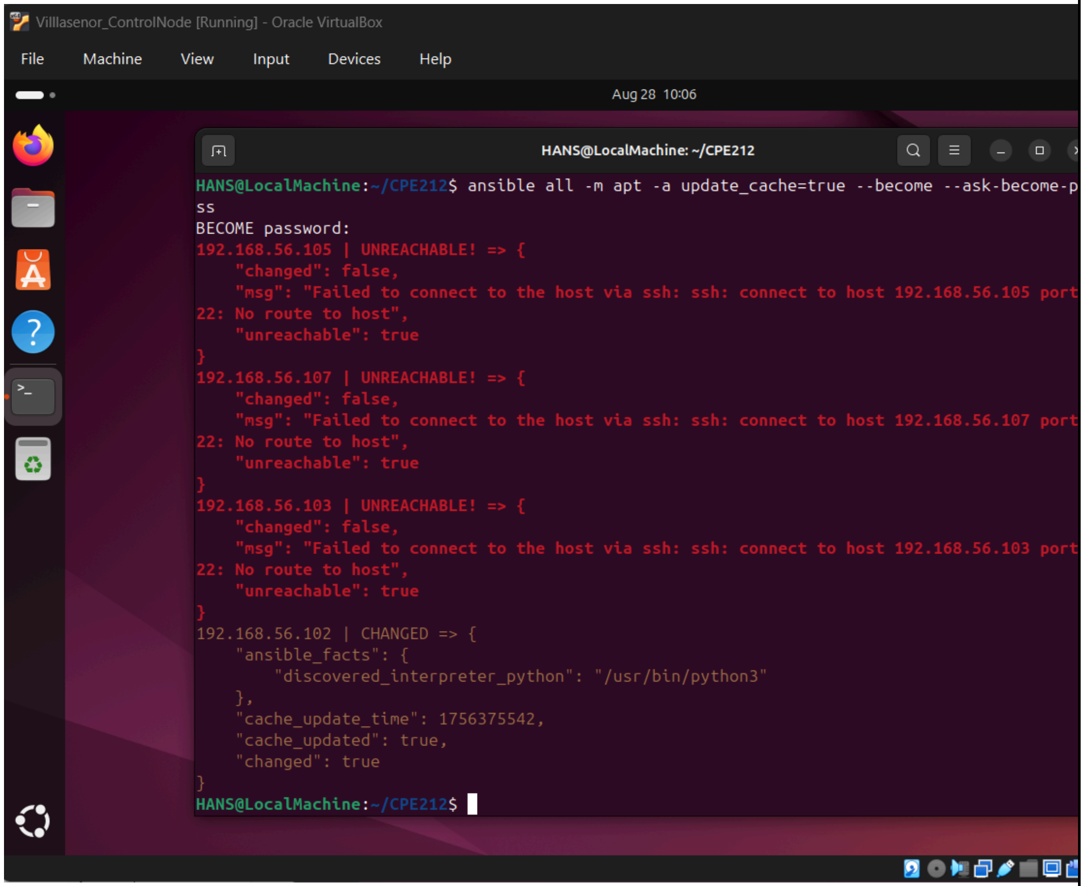
```
HANS@LocalMachine:~$ ansible all -m apt update_cache=true
usage: ansible [-h] [--version] [-v] [-b] [--become-method BECOME_METHOD]
               [--become-user BECOME_USER]
               [-K | --become-password-file BECOME_PASSWORD_FILE]
               [-i INVENTORY] [--list-hosts] [-l SUBSET] [-P POLL_INTERVAL]
               [-B SECONDS] [-o] [-t TREE] [--private-key PRIVATE_KEY_FILE]
               [-u REMOTE_USER] [-c CONNECTION] [-T TIMEOUT]
               [--ssh-common-args SSH_COMMON_ARGS]
               [--sftp-extra-args SFTP_EXTRA_ARGS]
               [--scp-extra-args SCP_EXTRA_ARGS]
               [--ssh-extra-args SSH_EXTRA_ARGS]
               [-k | --connection-password-file CONNECTION_PASSWORD_FILE] [-C]
               [-D] [-e EXTRA_VARS] [--vault-id VAULT_IDS]
               [-J | --vault-password-file VAULT_PASSWORD_FILES] [-f FORKS]
               [-M MODULE_PATH] [--playbook-dir BASEDIR]
               [--task-timeout TASK_TIMEOUT] [-a MODULE_ARGS] [-m MODULE_NAME]
               pattern
ansible: error: unrecognized arguments: update_cache=true

usage: ansible [-h] [--version] [-v] [-b] [--become-method BECOME_METHOD]
               [--become-user BECOME_USER]
               [-K | --become-password-file BECOME_PASSWORD_FILE]
               [-i INVENTORY] [--list-hosts] [-l SUBSET] [-P POLL_INTERVAL]
               [-B SECONDS] [-o] [-t TREE] [--private-key PRIVATE_KEY_FILE]
```

What is the result of the command? Is it successful?
Its not successful it just show the other commands.

Try editing the command and add something that would elevate the privilege. Issue the command *ansible all -m apt -a update_cache=true*

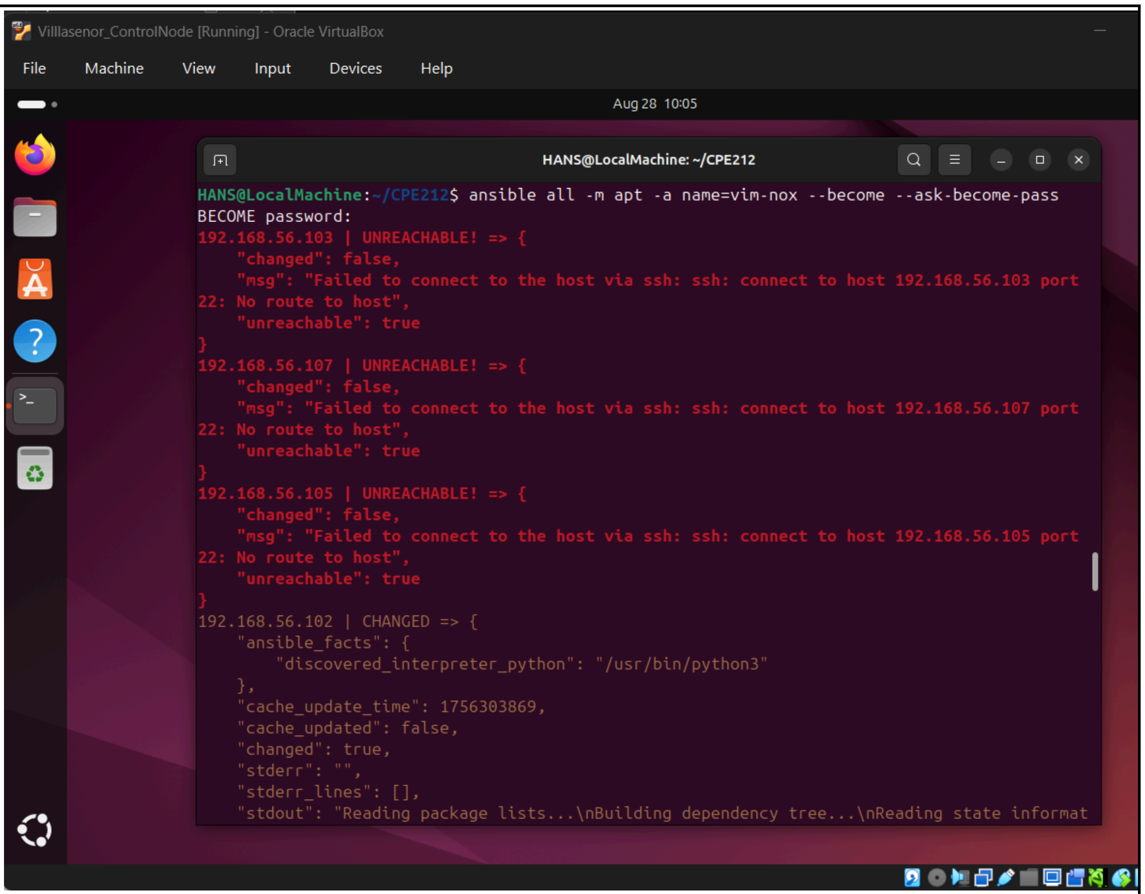
--become --ask-become-pass. Enter the sudo password when prompted. You will notice now that the output of this command is a success. The *update_cache=true* is the same thing as running *sudo apt update*. The --become command elevate the privileges and the *--ask-become-pass* asks for the password. For 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.


Villlasenor_ControlNode [Running] - Oracle VirtualBox
File
Machine
View
Input
Devices
Help
Aug 28 10:06
HANS@LocalMachine: ~/CPE212

```
HANS@LocalMachine:~/CPE212$ ansible all -m apt -a update_cache=true --become --ask-become-p
ss
BECOME password:
192.168.56.105 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.105 port
22: No route to host",
    "unreachable": true
}
192.168.56.107 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.107 port
22: No route to host",
    "unreachable": true
}
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
}
192.168.56.102 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1756375542,
    "cache_updated": true,
    "changed": true
}
HANS@LocalMachine:~/CPE212$
```

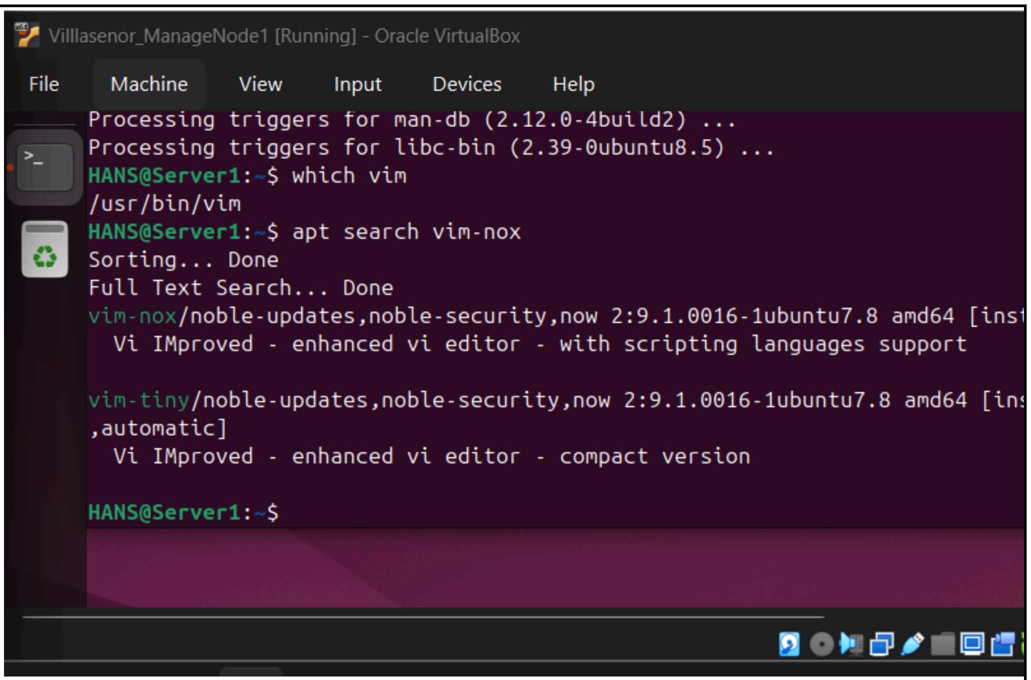
2. Let's try to install VIM, which is an almost compatible version of the UNIX editor Vi. To do this, we will just changed the module part in 1.1 instruction. Here is the command: *ansible all -m apt -a name=vim-nox --become --ask-become-pass.* The command would take some time after typing the password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


Villlasenor_ControlNode [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Aug 28 10:05
HANS@LocalMachine: ~/CPE212

```
HANS@LocalMachine:~/CPE212$ ansible all -m apt -a name=vim-nox --become --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
}
192.168.56.107 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.107 port
22: No route to host",
    "unreachable": true
}
192.168.56.105 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.105 port
22: No route to host",
    "unreachable": true
}
192.168.56.102 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1756303869,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading state informat
```

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?

Yes the command is successful because the vim is downloaded in the server

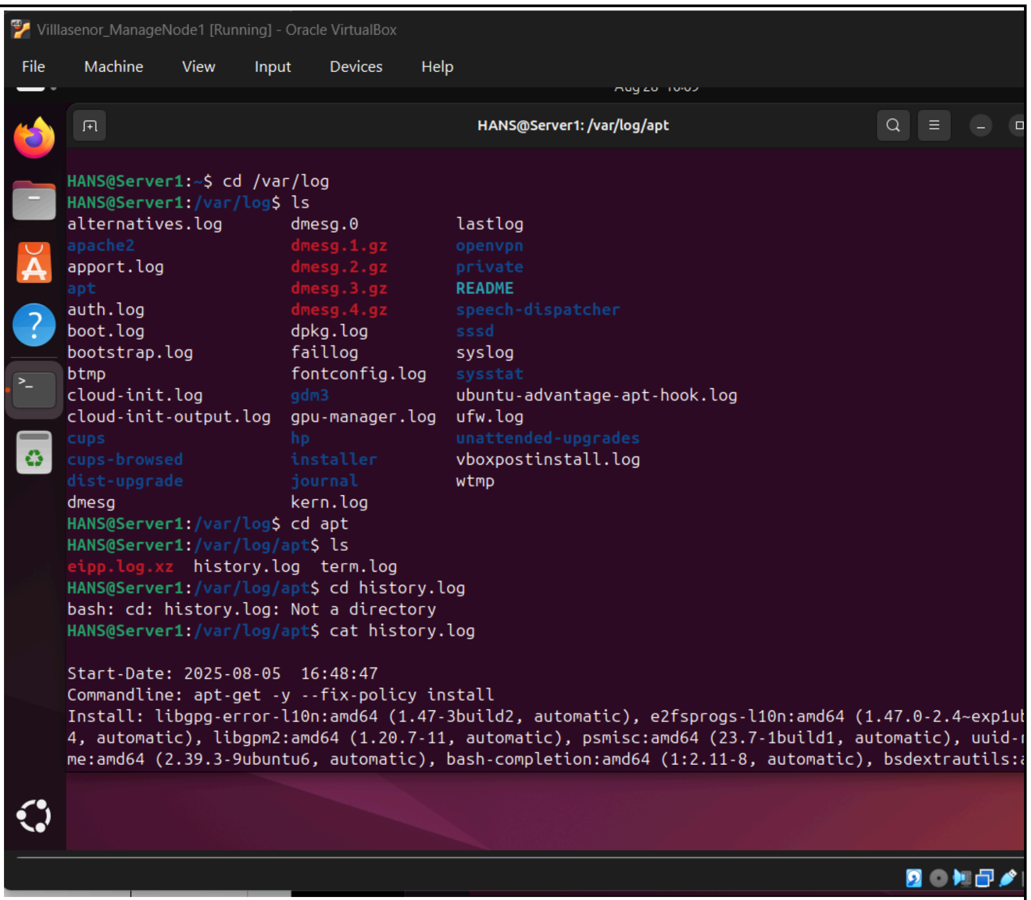


Villlasenor_ManageNode1 [Running] - Oracle VirtualBox
File Machine View Input Devices Help

```
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
HANS@Server1:~$ which vim
/usr/bin/vim
HANS@Server1:~$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/noble-updates,noble-security,now 2:9.1.0016-1ubuntu7.8 amd64 [inst
  Vi IMproved - enhanced vi editor - with scripting languages support

vim-tiny/noble-updates,noble-security,now 2:9.1.0016-1ubuntu7.8 amd64 [ins
,automatic]
  Vi IMproved - enhanced vi editor - compact version

HANS@Server1:~$
```

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.

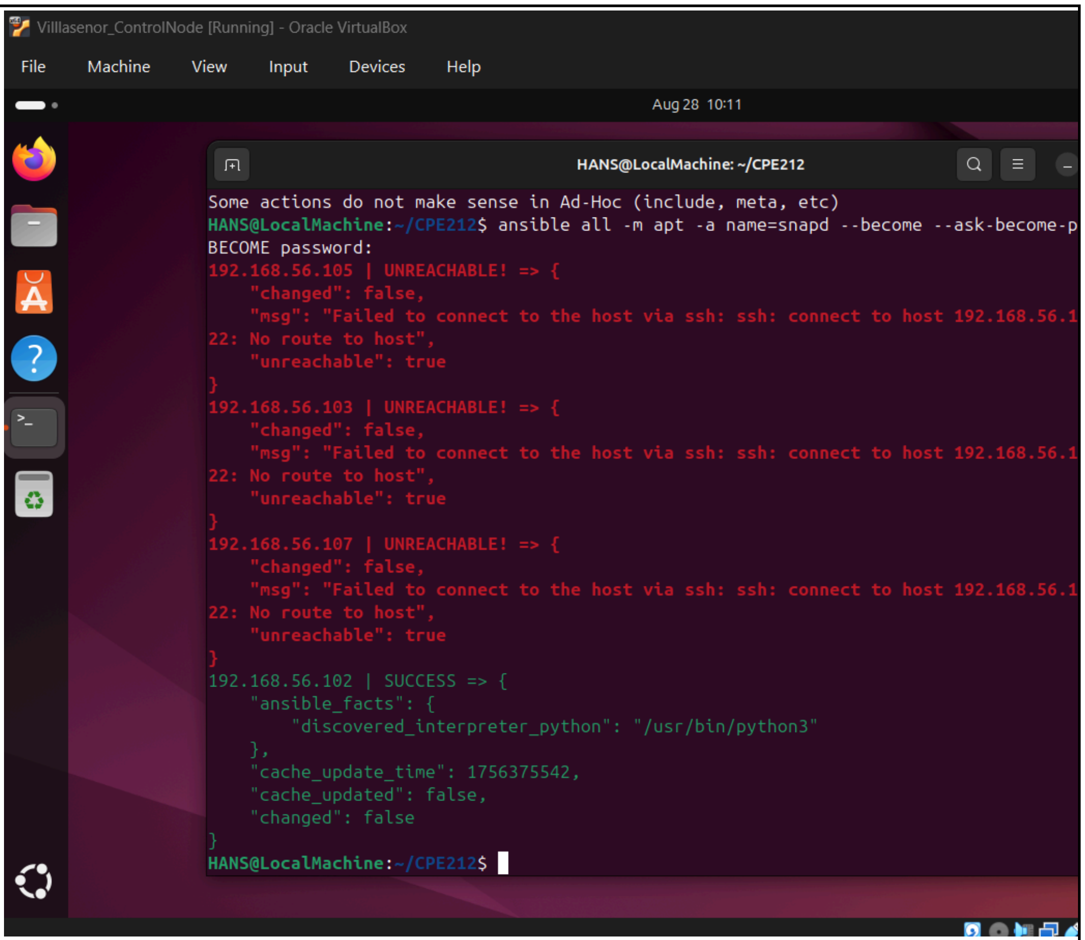
It shows the history po downloading the packages


Villlasenor_ManageNode1 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
HANS@Server1: /var/log/apt

```
HANS@Server1:~$ cd /var/log
HANS@Server1:/var/log$ ls
alternatives.log       dmesg.0           lastlog
apache2                dmesg.1.gz        openvpn
apport.log             dmesg.2.gz        private
apt                    dmesg.3.gz        README
auth.log               dmesg.4.gz        speech-dispatcher
boot.log               dpkg.log          sssd
bootstrap.log          faillog           syslog
btmp                   fontconfig.log    sysstat
cloud-init.log         gdm3              ubuntu-advantage-apt-hook.log
cloud-init-output.log  gpu-manager.log   ufw.log
cups                   hp                unattended-upgrades
cups-browsed           installer         vboxpostinstall.log
dist-upgrade           journal           wtmp
dmesg                  kern.log
HANS@Server1:/var/log$ cd apt
HANS@Server1:/var/log/apt$ ls
eipp.log.xz  history.log  term.log
HANS@Server1:/var/log/apt$ cd history.log
bash: cd: history.log: Not a directory
HANS@Server1:/var/log/apt$ cat history.log

Start-Date: 2025-08-05  16:48:47
Commandline: apt-get -y --fix-policy install
Install: libgpg-error-l10n:amd64 (1.47-3build2, automatic), e2fsprogs-l10n:amd64 (1.47.0-2.4~exp1ub
4, automatic), libgpm2:amd64 (1.20.7-11, automatic), psmisc:amd64 (23.7-1build1, automatic), uuid-r
me:amd64 (2.39.3-9ubuntu6, automatic), bash-completion:amd64 (1:2.11-8, automatic), bsdextrautils:a
```

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*



Villlasenor_ControlNode [Running] - Oracle VirtualBox
File
Machine
View
Input
Devices
Help
Aug 28 10:11
HANS@LocalMachine: ~/CPE212

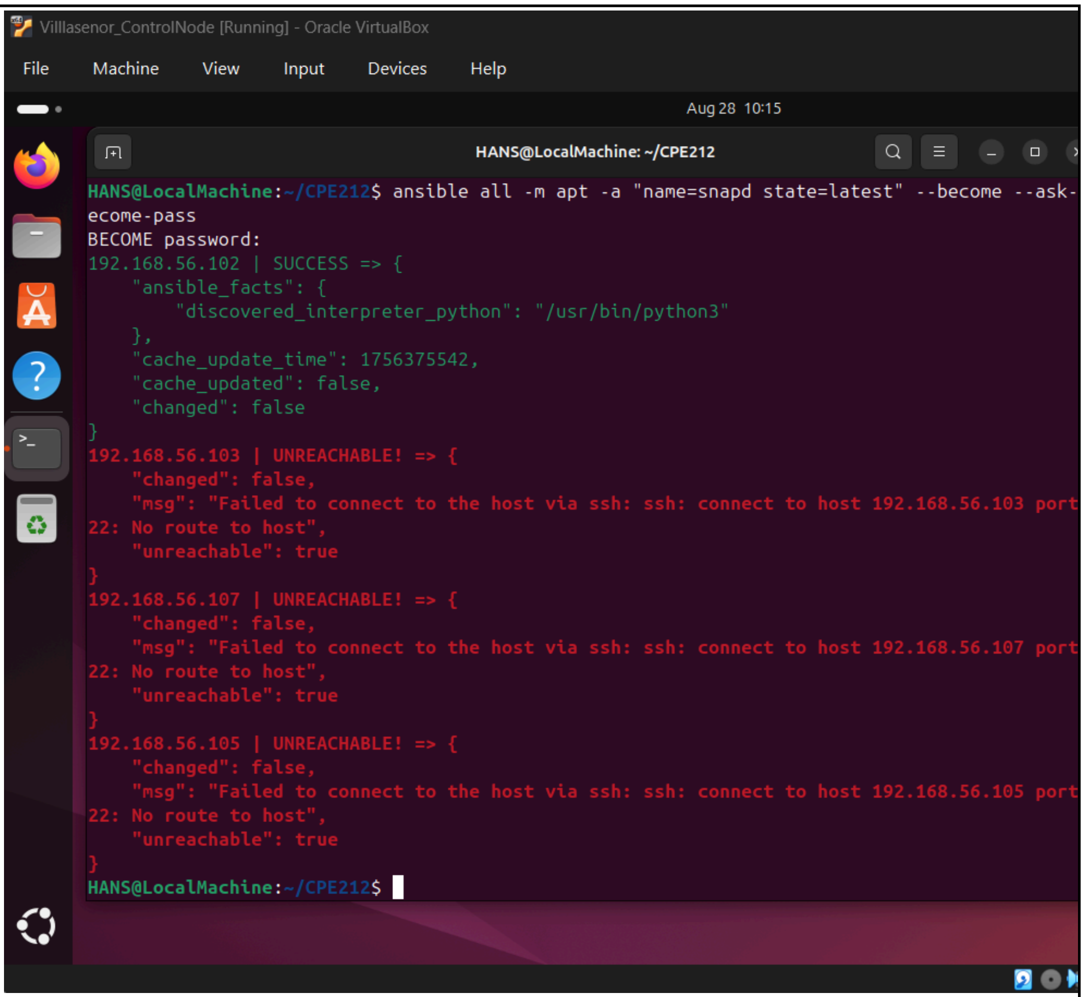
```
Some actions do not make sense in Ad-Hoc (include, meta, etc)
HANS@LocalMachine:~/CPE212$ ansible all -m apt -a name=snapd --become --ask-become-p
BECOME password:
192.168.56.105 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.1
22: No route to host",
    "unreachable": true
}
192.168.56.103 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.1
22: No route to host",
    "unreachable": true
}
192.168.56.107 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.1
22: No route to host",
    "unreachable": true
}
192.168.56.102 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1756375542,
    "cache_updated": false,
    "changed": false
}
HANS@LocalMachine:~/CPE212$
```

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

I think its successful because it reach the dbserver but it didnt not change in remote server

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.


Villlasenor_ControlNode [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Aug 28 10:15
HANS@LocalMachine: ~/CPE212

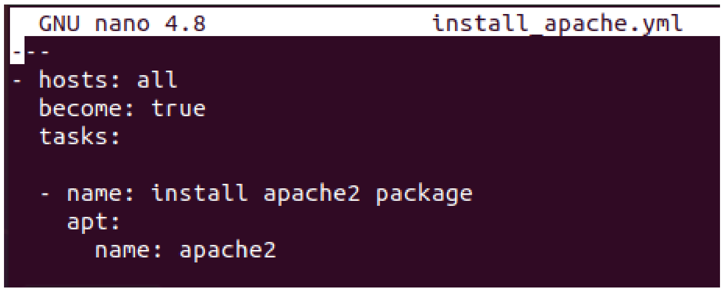
```
HANS@LocalMachine:~/CPE212$ ansible all -m apt -a "name=snapd state=latest" --become --ask-
ecome-pass
BECOME password:
192.168.56.102 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1756375542,
    "cache_updated": false,
    "changed": false
}
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
}
192.168.56.107 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.107 port
22: No route to host",
    "unreachable": true
}
192.168.56.105 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.105 port
22: No route to host",
    "unreachable": true
}
HANS@LocalMachine:~/CPE212$
```

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

Task 2: Writing our First Playbook

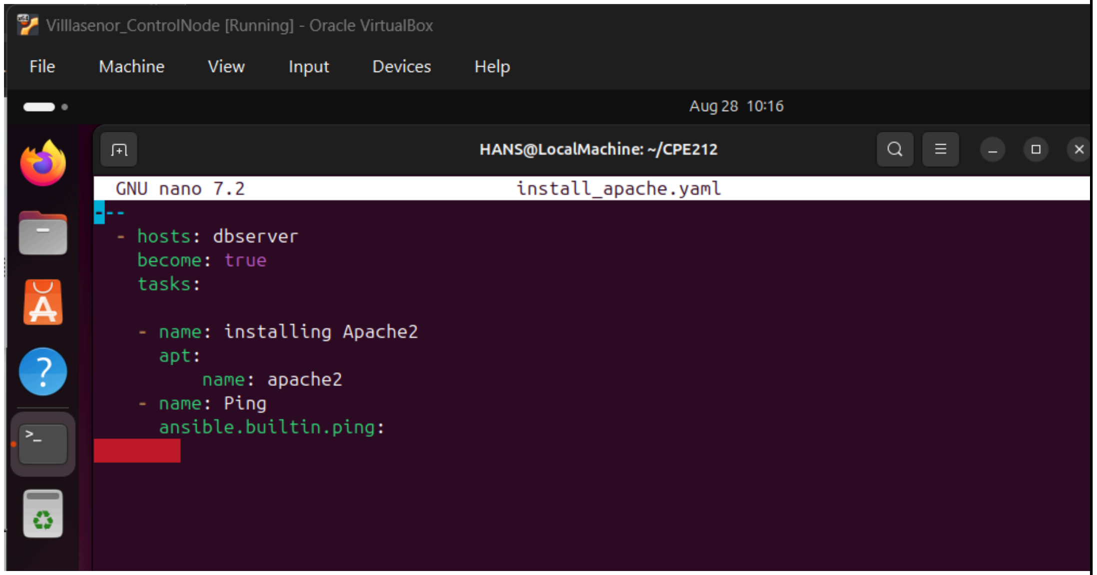
1. With ad hoc commands, we can simplify the administration of remote servers. For example, we can install updates, packages, and applications, etc. However, the real strength of ansible comes from its playbooks. When we write a playbook, we can define the state that we want our servers to be in and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



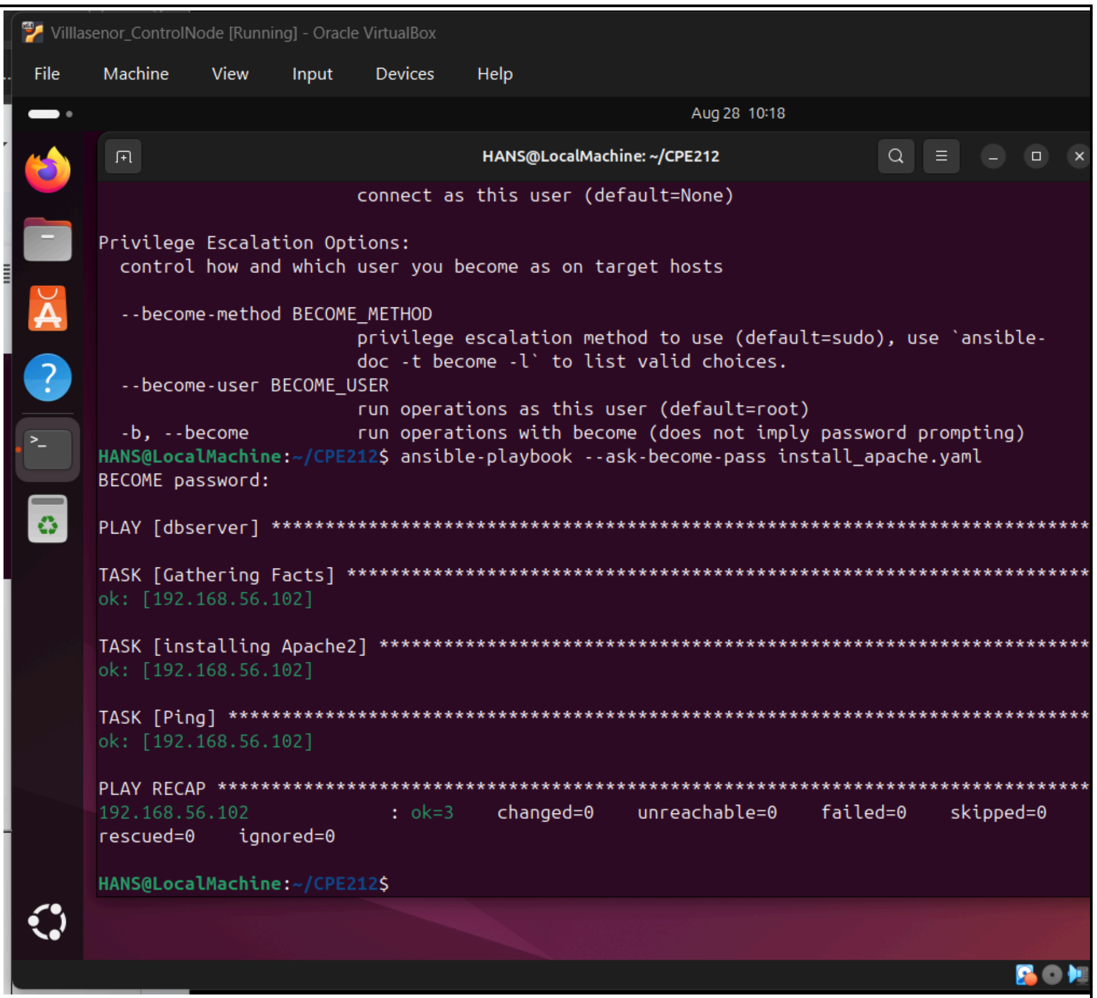
```
GNU nano 4.8                     install_apache.yml
---
- hosts: all
  become: true
  tasks:

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

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


Villlasenor_ControlNode [Running] - Oracle VirtualBox
File
Machine
View
Input
Devices
Help
Aug 28 10:16
HANS@LocalMachine: ~/CPE212

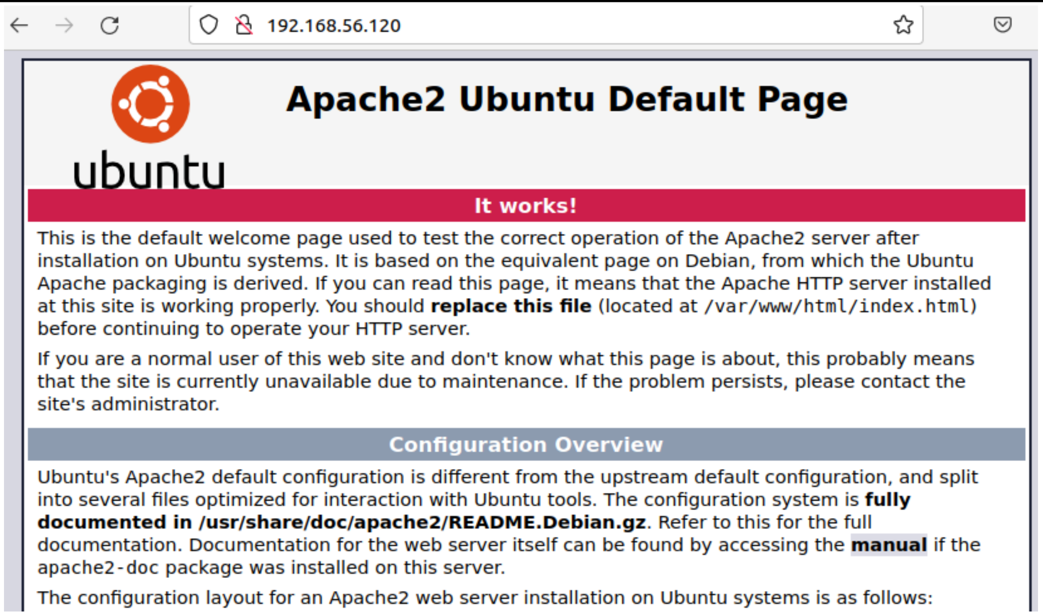
```
GNU nano 7.2                              install_apache.yaml
---
  - hosts: dbserver
    become: true
    tasks:

    - name: installing Apache2
      apt:
          name: apache2
    - name: Ping
      ansible.builtin.ping:
```

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


Villlasenor_ControlNode [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Aug 28 10:18
HANS@LocalMachine: ~/CPE212

```
                        connect as this user (default=None)

Privilege Escalation Options:
  control how and which user you become as on target hosts

  --become-method BECOME_METHOD
                        privilege escalation method to use (default=sudo), use `ansible-
                        doc -t become -l` to list valid choices.
  --become-user BECOME_USER
                        run operations as this user (default=root)
  -b, --become          run operations with become (does not imply password prompting)
HANS@LocalMachine:~/CPE212$ ansible-playbook --ask-become-pass install_apache.yaml
BECOME password:

PLAY [dbserver] ****************************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [192.168.56.102]

TASK [installing Apache2] ******************************************************************
ok: [192.168.56.102]

TASK [Ping] ********************************************************************************
ok: [192.168.56.102]

PLAY RECAP *********************************************************************************
192.168.56.102             : ok=3    changed=0    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0

HANS@LocalMachine:~/CPE212$
```

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.

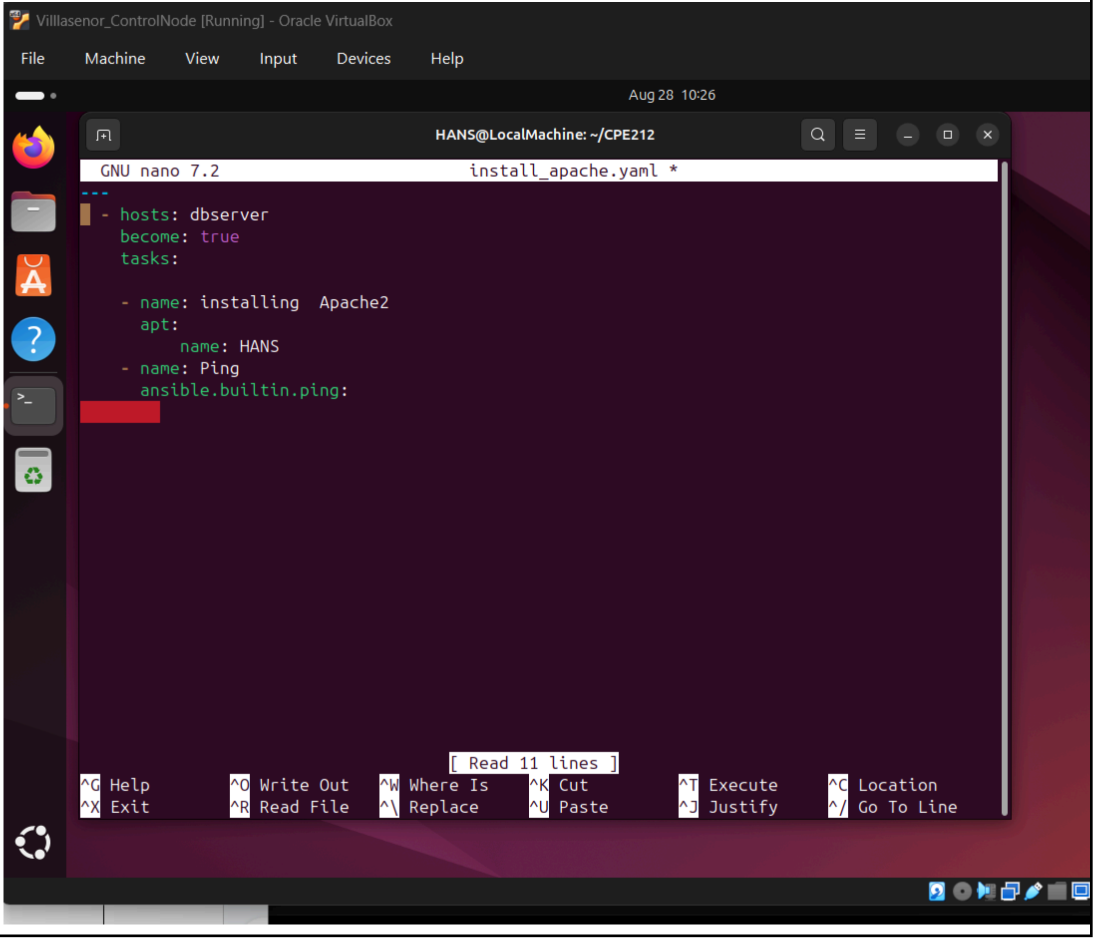
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

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.

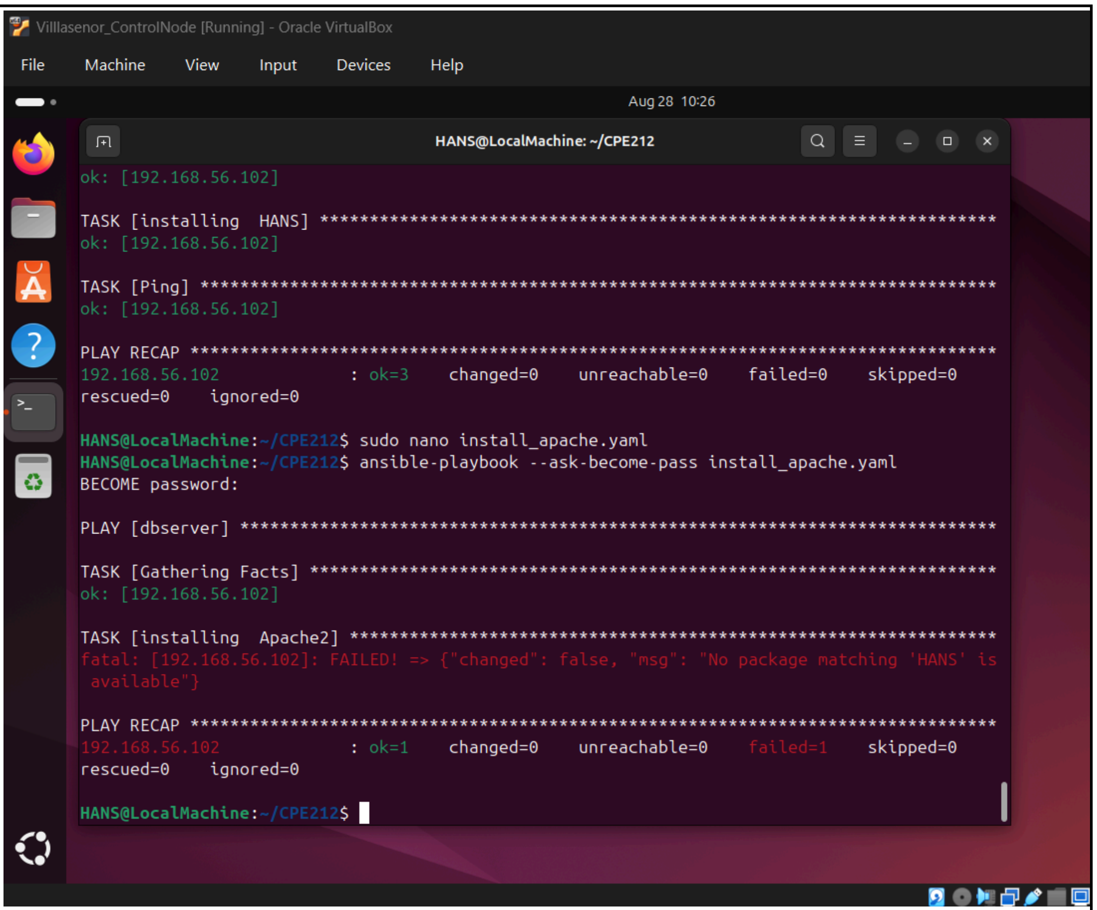
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

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?



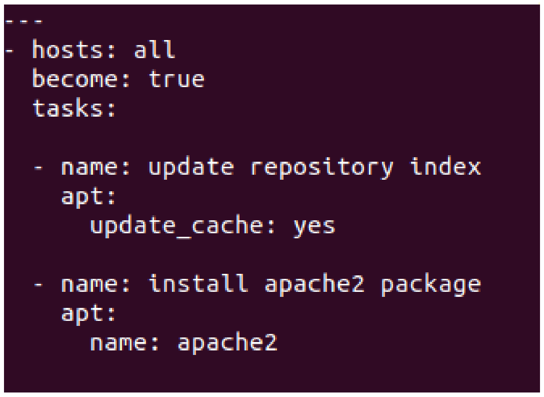
Villlasenor_ControlNode [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Aug 28 10:26
HANS@LocalMachine: ~/CPE212

```
GNU nano 7.2                         install_apache.yaml *
---
- hosts: dbserver
  become: true
  tasks:

  - name: installing  Apache2
    apt:
      name: HANS
  - name: Ping
    ansible.builtin.ping:

                                [ Read 11 lines ]
^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

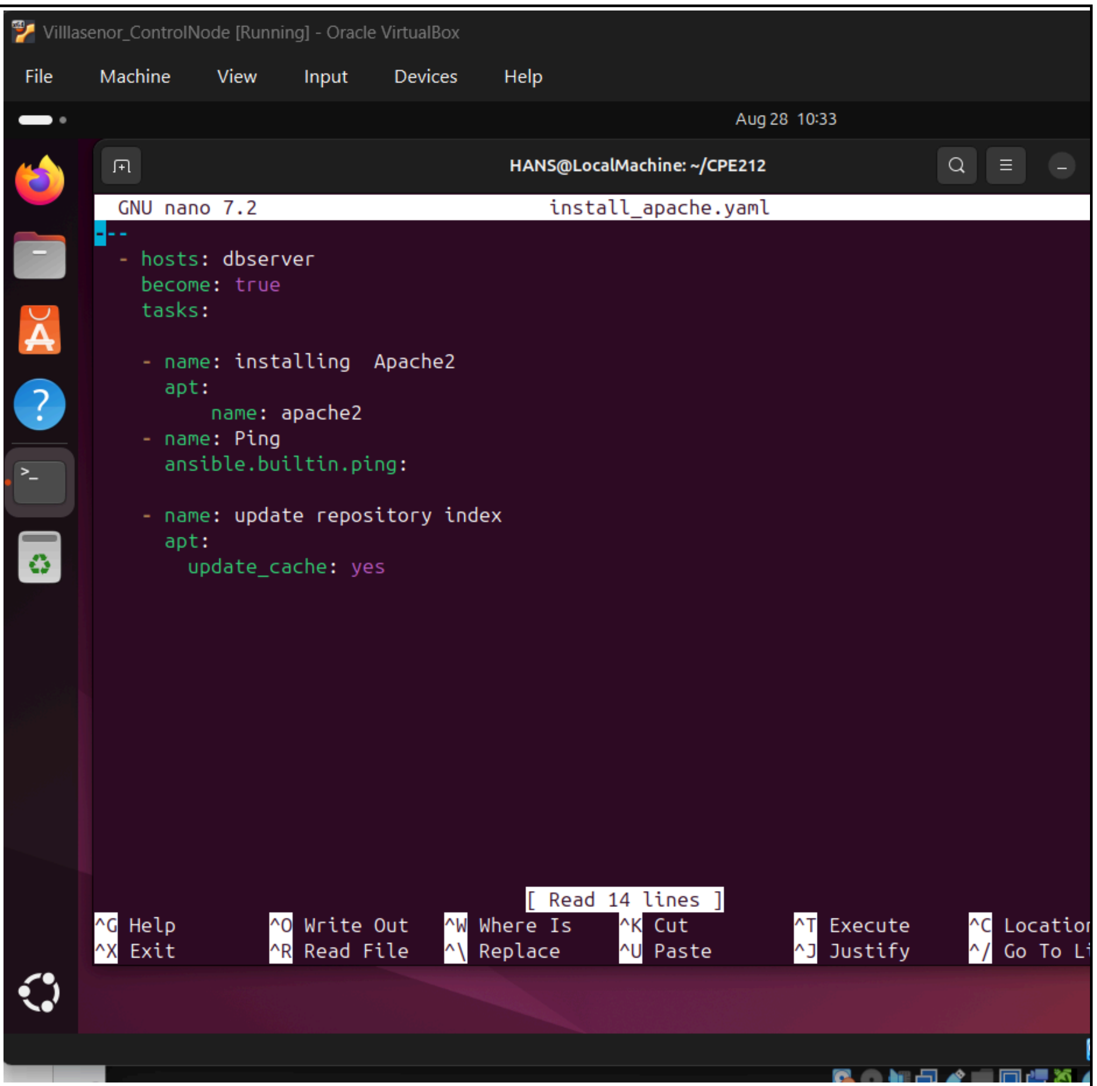

Villasenor_ControlNode [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Aug 28 10:26
HANS@LocalMachine: ~/CPE212

```
ok: [192.168.56.102]

TASK [installing  HANS] *********************************************************************
ok: [192.168.56.102]

TASK [Ping] *********************************************************************************
ok: [192.168.56.102]

PLAY RECAP **********************************************************************************
192.168.56.102             : ok=3    changed=0    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0

HANS@LocalMachine:~/CPE212$ sudo nano install_apache.yaml
HANS@LocalMachine:~/CPE212$ ansible-playbook --ask-become-pass install_apache.yaml
BECOME password:

PLAY [dbserver] *****************************************************************************

TASK [Gathering Facts] **********************************************************************
ok: [192.168.56.102]

TASK [installing  Apache2] ******************************************************************
fatal: [192.168.56.102]: FAILED! => {"changed": false, "msg": "No package matching 'HANS' is
 available"}

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

HANS@LocalMachine:~/CPE212$
```

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
```

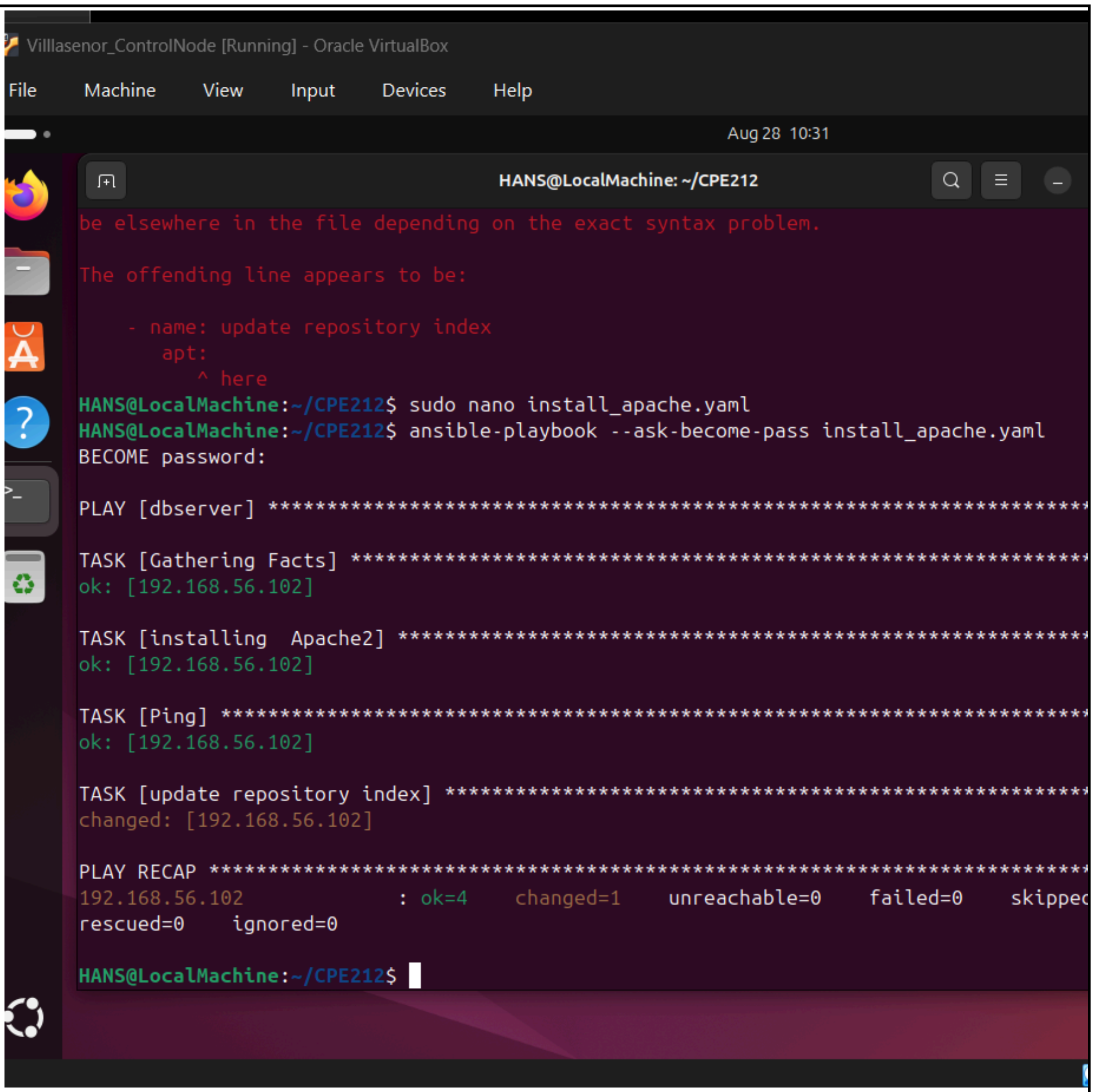

Villlasenor_ControlNode [Running] - Oracle VirtualBox

HANS@LocalMachine: ~/CPE212

```
GNU nano 7.2                        install_apache.yaml
---
- hosts: dbserver
  become: true
  tasks:

  - name: installing  Apache2
    apt:
        name: apache2
  - name: Ping
    ansible.builtin.ping:

  - name: update repository index
    apt:
      update_cache: yes




                                     [ Read 14 lines ]
^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Locatior
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Li
```

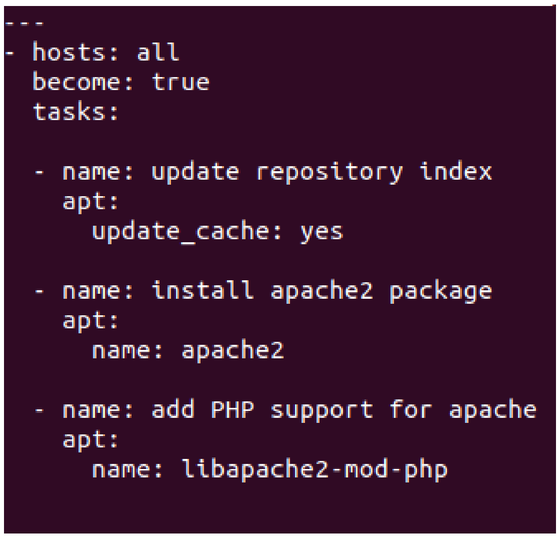
Save the changes to this file and exit.

6. Run the playbook and describe the output. Did the new command change anything on the remote servers? **Yes, it change in the task repository index**


Villlasenor_ControlNode [Running] - Oracle VirtualBox
File
Machine
View
Input
Devices
Help
Aug 28 10:31
HANS@LocalMachine: ~/CPE212

```
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

    - name: update repository index
      apt:
         ^ here
HANS@LocalMachine:~/CPE212$ sudo nano install_apache.yaml
HANS@LocalMachine:~/CPE212$ ansible-playbook --ask-become-pass install_apache.yaml
BECOME password:

PLAY [dbserver] **********************************************************************

TASK [Gathering Facts] ***************************************************************
ok: [192.168.56.102]

TASK [installing  Apache2] ***********************************************************
ok: [192.168.56.102]

TASK [Ping] **************************************************************************
ok: [192.168.56.102]

TASK [update repository index] *******************************************************
changed: [192.168.56.102]

PLAY RECAP ***************************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0    skippec
rescued=0    ignored=0

HANS@LocalMachine:~/CPE212$ 
```

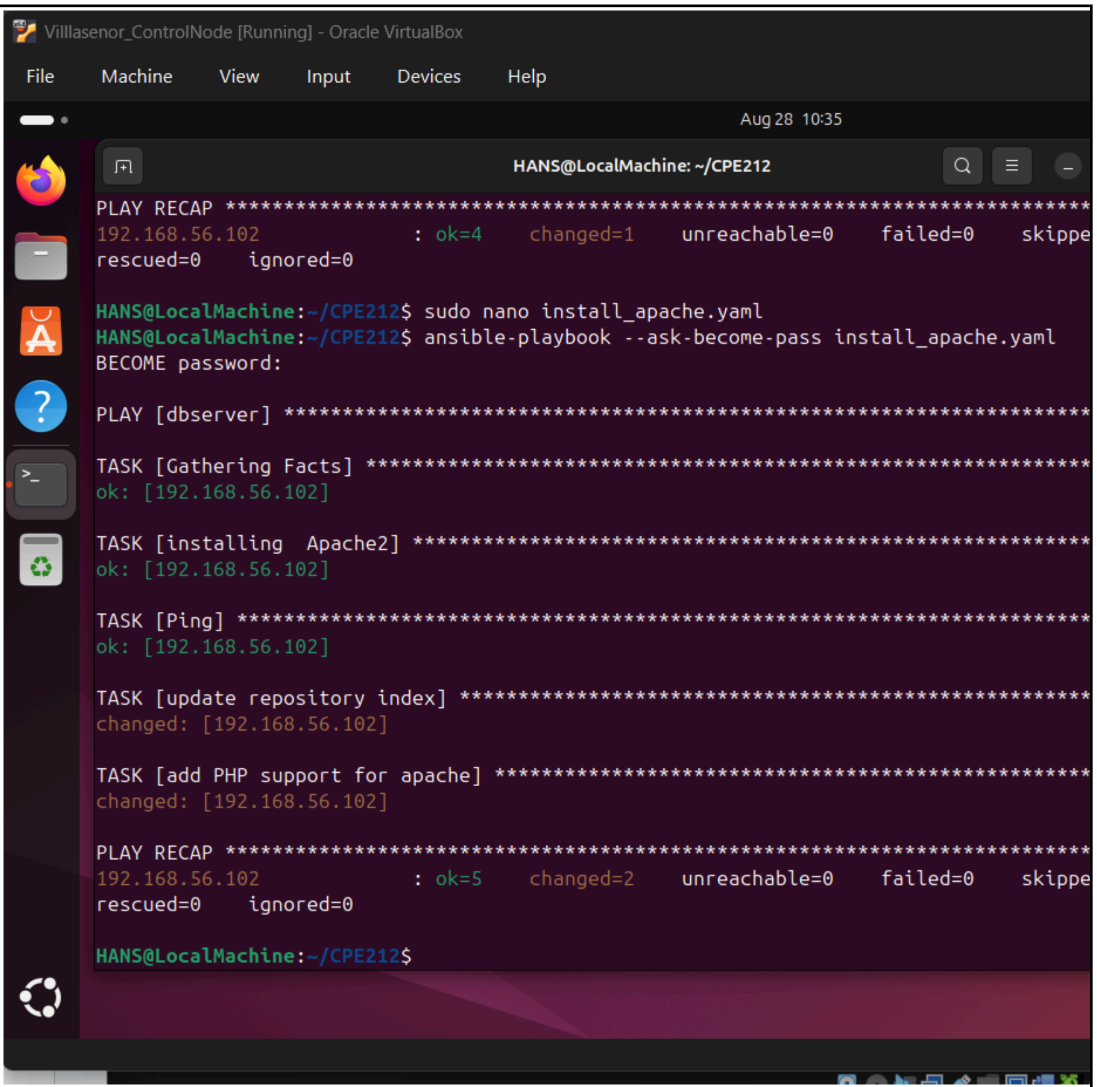
7. Edit again the *install_apache.yml*. This time, we are going to add a PHP support for the apache package we installed earlier.



```
---
- 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?


Villlasenor_ControlNode [Running] - Oracle VirtualBox
File
Machine
View
Input
Devices
Help
Aug 28 10:35
HANS@LocalMachine: ~/CPE212

```
PLAY RECAP *********************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0    skippe
rescued=0    ignored=0

HANS@LocalMachine:~/CPE212$ sudo nano install_apache.yaml
HANS@LocalMachine:~/CPE212$ ansible-playbook --ask-become-pass install_apache.yaml
BECOME password:

PLAY [dbserver] ****************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.102]

TASK [installing  Apache2] *****************************************************
ok: [192.168.56.102]

TASK [Ping] ********************************************************************
ok: [192.168.56.102]

TASK [update repository index] *************************************************
changed: [192.168.56.102]

TASK [add PHP support for apache] **********************************************
changed: [192.168.56.102]

PLAY RECAP *********************************************************************
192.168.56.102             : ok=5    changed=2    unreachable=0    failed=0    skippe
rescued=0    ignored=0

HANS@LocalMachine:~/CPE212$
```

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

   https://github.com/hrvillasenor-ux/CPE31S2_VILLASENOR

Reflections:

Answer the following:

1. What is the importance of using a playbook?
   A playbook is important because it helps automate Ansible tasks. It makes it easier to manage updates, installations, and other changes on many machines at once, while keeping everything consistent.
2. Summarize what we have done on this activity.

In this activity, I used ad hoc commands in Ansible to install, update, and upgrade packages on remote machines. I also created a playbook to automate these tasks, which made the work faster and easier to repeat.