Name: Valencia, Mark Janssen	**Date Performed: September 11, 2023**
Course/Section: Cpe31s6	**Date Submitted: September 11, 2023**
Instructor: Dr. Jonathan Taylar	**Semester and SY: 1st Sem, 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 makes changes to the remote servers. We manage to gather facts and connect to the remote machines, but we still did not make changes on those machines. In this activity, we will learn to use commands that would install, update, and upgrade packages in the remote machines. We will also create a playbook that will be used for automations.

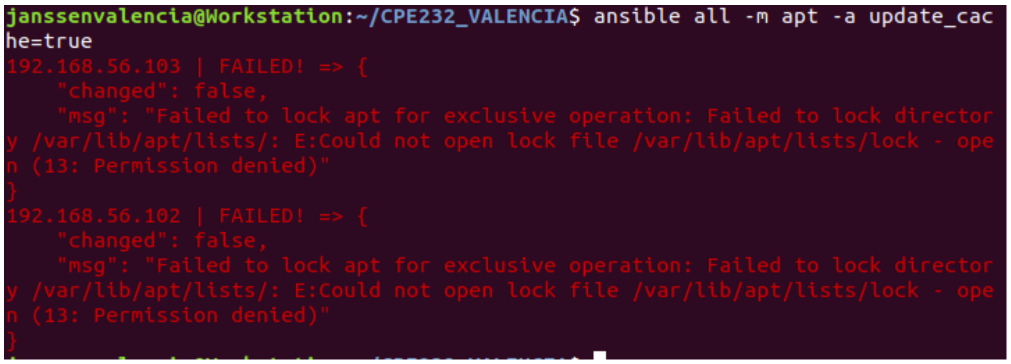
Playbooks record and execute **Ansible**'s configuration, deployment, and orchestration functions. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. If Ansible modules are the tools in your workshop, playbooks are your instruction manuals,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

Task 1: Run elevated ad hoc commands

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

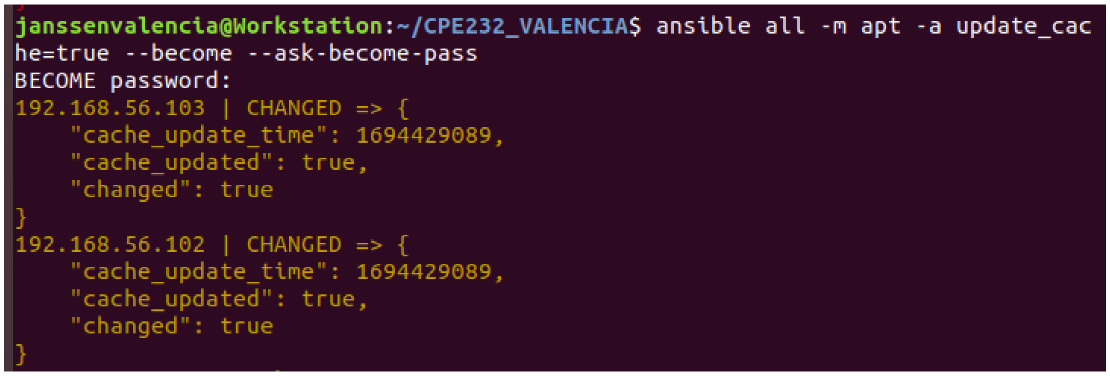
ansible all -m apt -a update_cache=true

What is the result of the command? Is it successful? **No**



```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible all -m apt -a update_cac
he=true
192.168.56.103 | FAILED! => {
    "changed": false,
    "msg": "Failed to lock apt for exclusive operation: Failed to lock director
y /var/lib/apt/lists/: E:Could not open lock file /var/lib/apt/lists/lock - ope
n (13: Permission denied)"
}
192.168.56.102 | FAILED! => {
    "changed": false,
    "msg": "Failed to lock apt for exclusive operation: Failed to lock director
y /var/lib/apt/lists/: E:Could not open lock file /var/lib/apt/lists/lock - ope
n (13: Permission denied)"
}
```

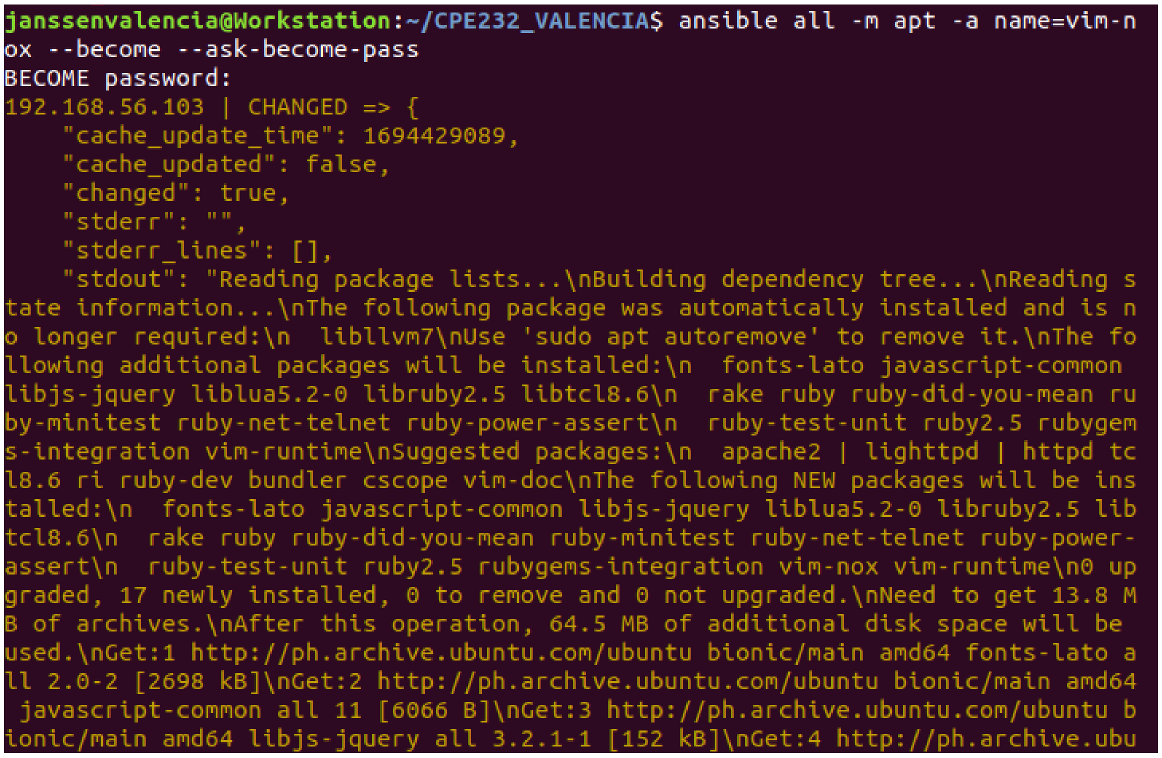
Try editing the command and add something that would elevate the privilege. Issue the command *ansible all -m apt -a update_cache=true --become --ask-become-pass.* Enter the sudo password when prompted. You will notice now that the output of this command is a success. The *update_cache=true* is the same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



```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible all -m apt -a update_cac
he=true --become --ask-become-pass
BECOME password:
192.168.56.103 | CHANGED => {
    "cache_update_time": 1694429089,
    "cache_updated": true,
    "changed": true
}
192.168.56.102 | CHANGED => {
    "cache_update_time": 1694429089,
    "cache_updated": true,
    "changed": true
}
```

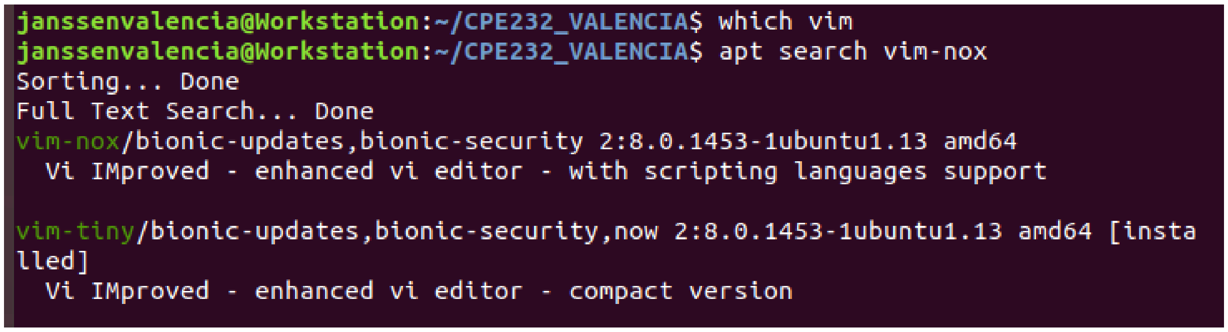
You may notice after the second command was executed, the status is CHANGED compared to the first command, which is FAILED.

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



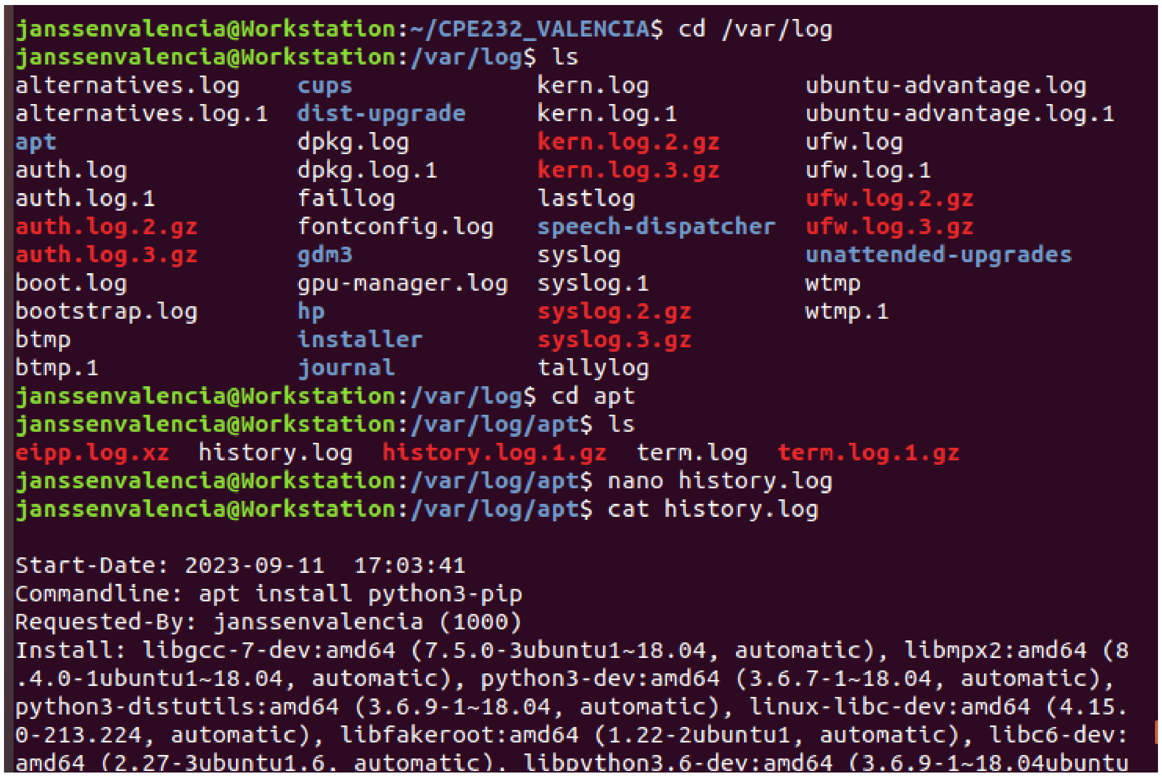
```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible all -m apt -a name=vim-n
ox --become --ask-become-pass
BECOME password:
192.168.56.103 | CHANGED => {
    "cache_update_time": 1694429089,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading s
tate information...\nThe following package was automatically installed and is n
o longer required:\n  libllvm7\nUse 'sudo apt autoremove' to remove it.\nThe fo
llowing additional packages will be installed:\n  fonts-lato javascript-common
libjs-jquery liblua5.2-0 libruby2.5 libtcl8.6\n  rake ruby ruby-did-you-mean ru
by-minitest ruby-net-telnet ruby-power-assert\n  ruby-test-unit ruby2.5 rubygem
s-integration vim-runtime\nSuggested packages:\n  apache2 | lighttpd | httpd tc
l8.6 ri ruby-dev bundler cscope vim-doc\nThe following NEW packages will be ins
talled:\n  fonts-lato javascript-common libjs-jquery liblua5.2-0 libruby2.5 lib
tcl8.6\n  rake ruby ruby-did-you-mean ruby-minitest ruby-net-telnet ruby-power-
assert\n  ruby-test-unit ruby2.5 rubygems-integration vim-nox vim-runtime\n0 up
graded, 17 newly installed, 0 to remove and 0 not upgraded.\nNeed to get 13.8 M
B of archives.\nAfter this operation, 64.5 MB of additional disk space will be
used.\nGet:1 http://ph.archive.ubuntu.com/ubuntu bionic/main amd64 fonts-lato a
ll 2.0-2 [2698 kB]\nGet:2 http://ph.archive.ubuntu.com/ubuntu bionic/main amd64
 javascript-common all 11 [6066 B]\nGet:3 http://ph.archive.ubuntu.com/ubuntu b
ionic/main amd64 libjs-jquery all 3.2.1-1 [152 kB]\nGet:4 http://ph.archive.ubu
```

1.1 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?

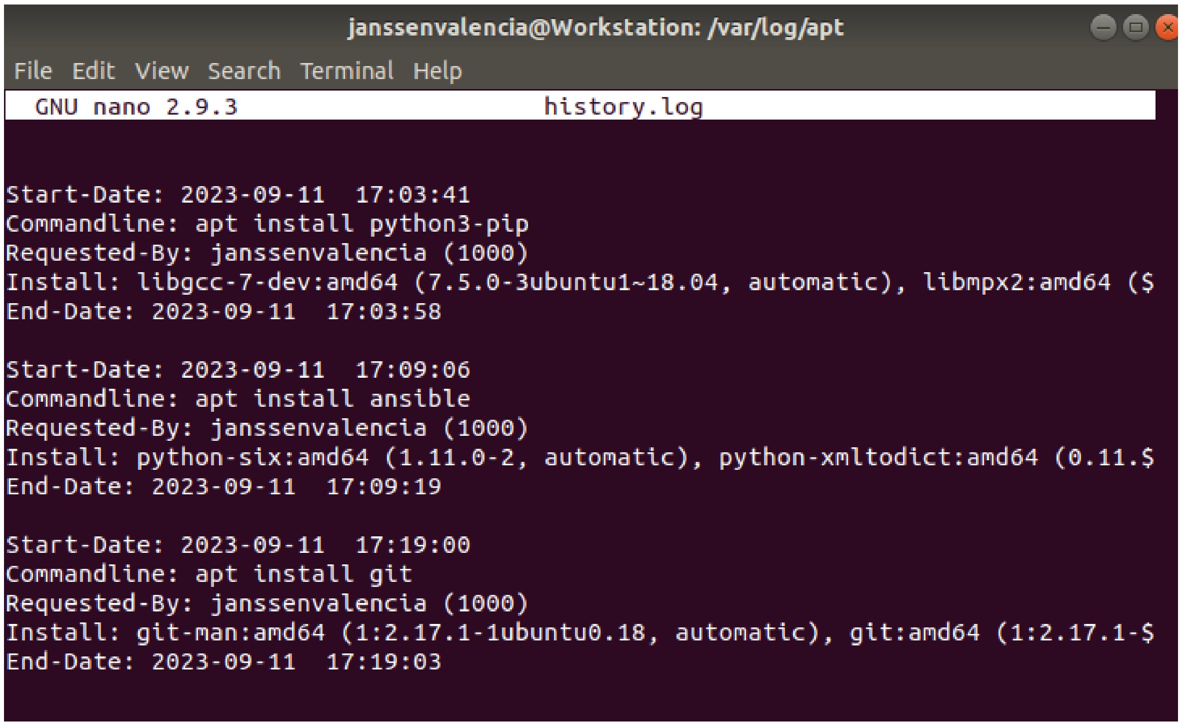


```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ which vim
janssenvalencia@Workstation:~/CPE232_VALENCIA$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/bionic-updates,bionic-security 2:8.0.1453-1ubuntu1.13 amd64
  Vi IMproved - enhanced vi editor - with scripting languages support

vim-tiny/bionic-updates,bionic-security,now 2:8.0.1453-1ubuntu1.13 amd64 [insta
lled]
  Vi IMproved - enhanced vi editor - compact version
```

1.2 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.



```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ cd /var/log
janssenvalencia@Workstation:/var/log$ ls
alternatives.log    cups              kern.log           ubuntu-advantage.log
alternatives.log.1  dist-upgrade      kern.log.1         ubuntu-advantage.log.1
apt                 dpkg.log          kern.log.2.gz      ufw.log
auth.log            dpkg.log.1        kern.log.3.gz      ufw.log.1
auth.log.1          faillog           lastlog            ufw.log.2.gz
auth.log.2.gz       fontconfig.log    speech-dispatcher  ufw.log.3.gz
auth.log.3.gz       gdm3              syslog             unattended-upgrades
boot.log            gpu-manager.log   syslog.1           wtmp
bootstrap.log       hp                syslog.2.gz        wtmp.1
btmp                installer         syslog.3.gz
btmp.1              journal           tallylog
janssenvalencia@Workstation:/var/log$ cd apt
janssenvalencia@Workstation:/var/log/apt$ ls
eipp.log.xz  history.log  history.log.1.gz  term.log  term.log.1.gz
janssenvalencia@Workstation:/var/log/apt$ nano history.log
janssenvalencia@Workstation:/var/log/apt$ cat history.log

Start-Date: 2023-09-11  17:03:41
Commandline: apt install python3-pip
Requested-By: janssenvalencia (1000)
Install: libgcc-7-dev:amd64 (7.5.0-3ubuntu1~18.04, automatic), libmpx2:amd64 (8
.4.0-1ubuntu1~18.04, automatic), python3-dev:amd64 (3.6.7-1~18.04, automatic),
python3-distutils:amd64 (3.6.9-1~18.04, automatic), linux-libc-dev:amd64 (4.15.
0-213.224, automatic), libfakeroot:amd64 (1.22-2ubuntu1, automatic), libc6-dev:
amd64 (2.27-3ubuntu1.6, automatic), libpython3.6-dev:amd64 (3.6.9-1~18.04ubuntu
```

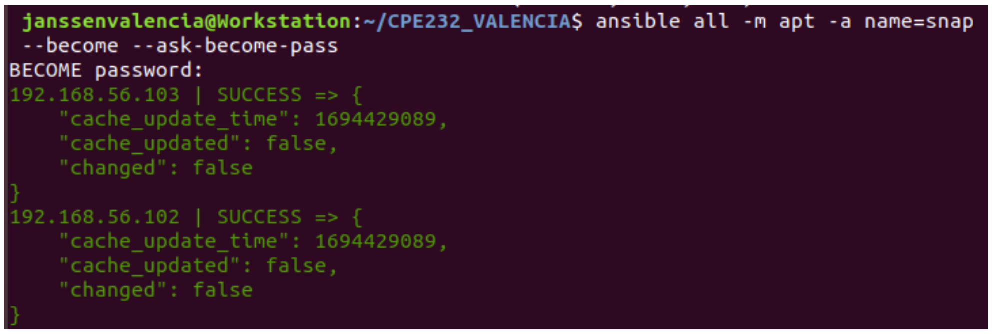


janssenvalencia@Workstation: /var/log/apt
File Edit View Search Terminal Help

```
  GNU nano 2.9.3                      history.log


Start-Date: 2023-09-11  17:03:41
Commandline: apt install python3-pip
Requested-By: janssenvalencia (1000)
Install: libgcc-7-dev:amd64 (7.5.0-3ubuntu1~18.04, automatic), libmpx2:amd64 ($
End-Date: 2023-09-11  17:03:58

Start-Date: 2023-09-11  17:09:06
Commandline: apt install ansible
Requested-By: janssenvalencia (1000)
Install: python-six:amd64 (1.11.0-2, automatic), python-xmltodict:amd64 (0.11.$
End-Date: 2023-09-11  17:09:19

Start-Date: 2023-09-11  17:19:00
Commandline: apt install git
Requested-By: janssenvalencia (1000)
Install: git-man:amd64 (1:2.17.1-1ubuntu0.18, automatic), git:amd64 (1:2.17.1-$
End-Date: 2023-09-11  17:19:03
```

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.

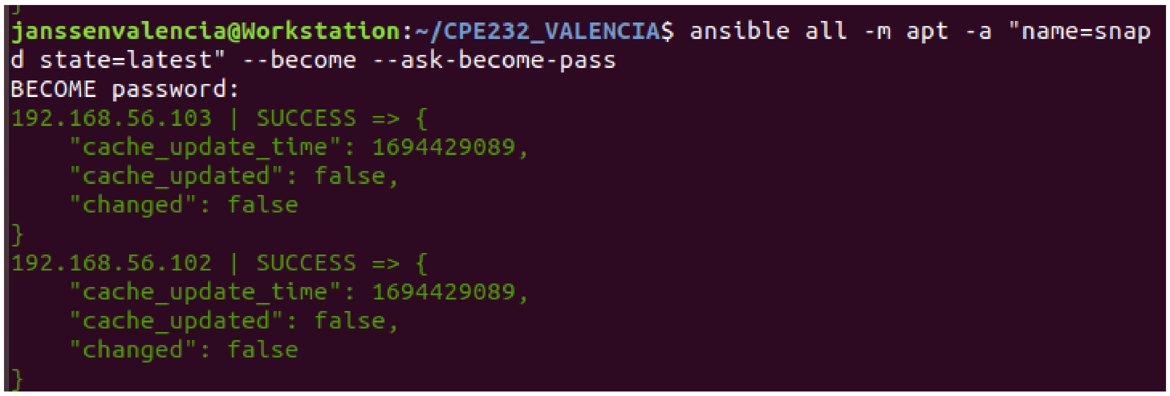
   3.1 Issue the command: *ansible all -m apt -a name=snapd --become --ask-become-pass*



```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible all -m apt -a name=snap
 --become --ask-become-pass
BECOME password:
192.168.56.103 | SUCCESS => {
    "cache_update_time": 1694429089,
    "cache_updated": false,
    "changed": false
}
192.168.56.102 | SUCCESS => {
    "cache_update_time": 1694429089,
    "cache_updated": false,
    "changed": false
}
```

   Can you describe the result of this command? Is it a success? Did it change anything in the remote servers? **Yes, the cache has been updated on both servers**

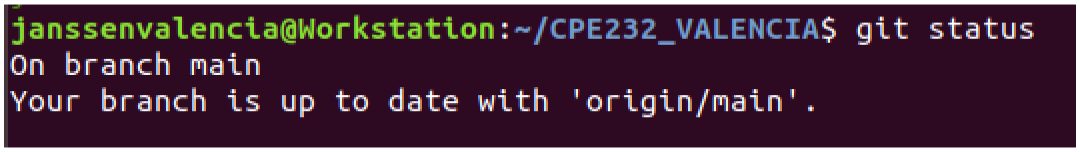
   3.2 Now, try to issue this command: *ansible all -m apt -a "name=snapd state=latest" --become --ask-become-pass*



```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible all -m apt -a "name=snap
d state=latest" --become --ask-become-pass
BECOME password:
192.168.56.103 | SUCCESS => {
    "cache_update_time": 1694429089,
    "cache_updated": false,
    "changed": false
}
192.168.56.102 | SUCCESS => {
    "cache_update_time": 1694429089,
    "cache_updated": false,
    "changed": false
}
```

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

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

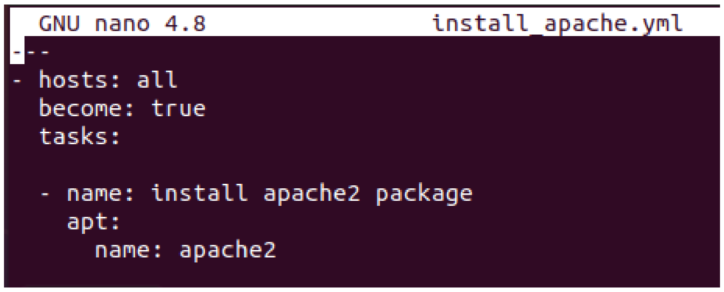


```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

Task 2: Writing our First Playbook

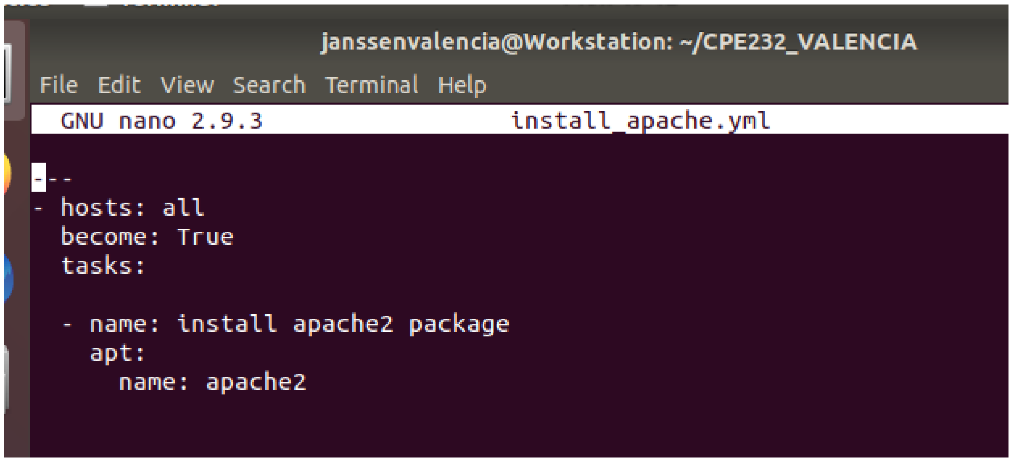
1. With ad hoc commands, we can simplify the administration of remote servers. For example, we can install updates, packages, and applications, etc. However, the real strength of ansible comes from its playbooks. When we write a playbook, we can define the state that we want our servers to be in and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



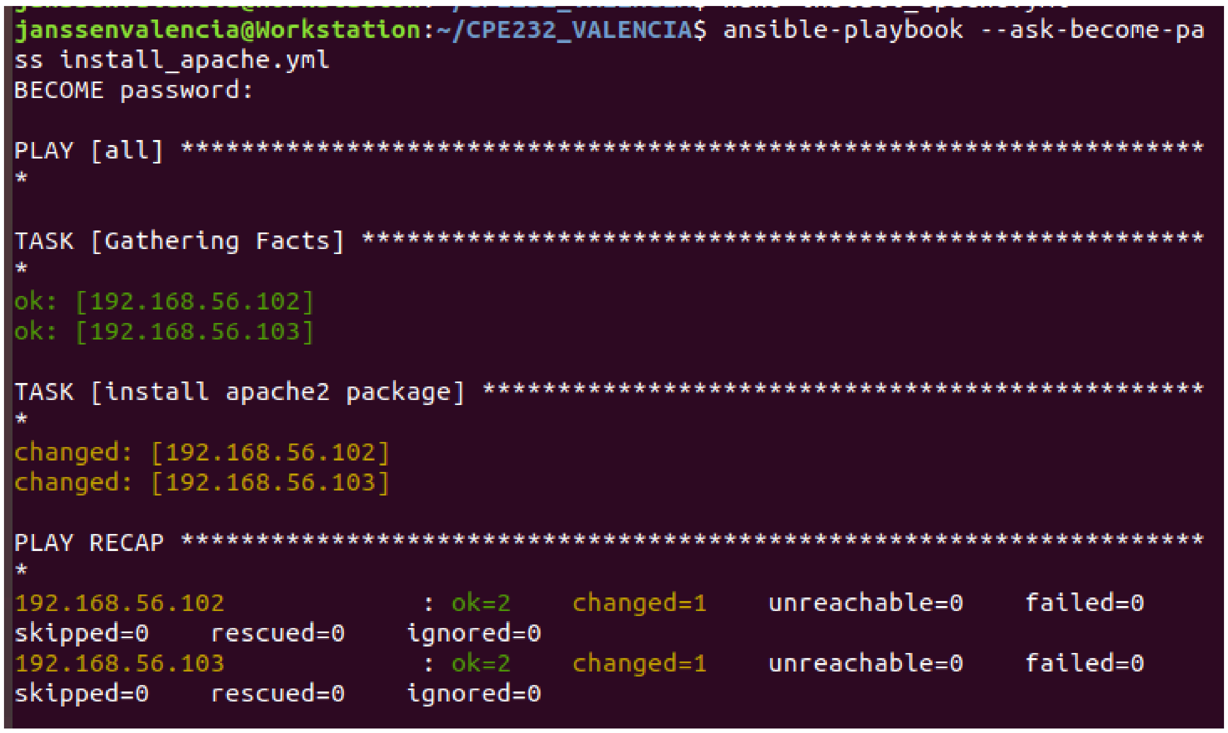
```
  GNU nano 4.8                      install_apache.yml
---
- hosts: all
  become: true
  tasks:

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

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



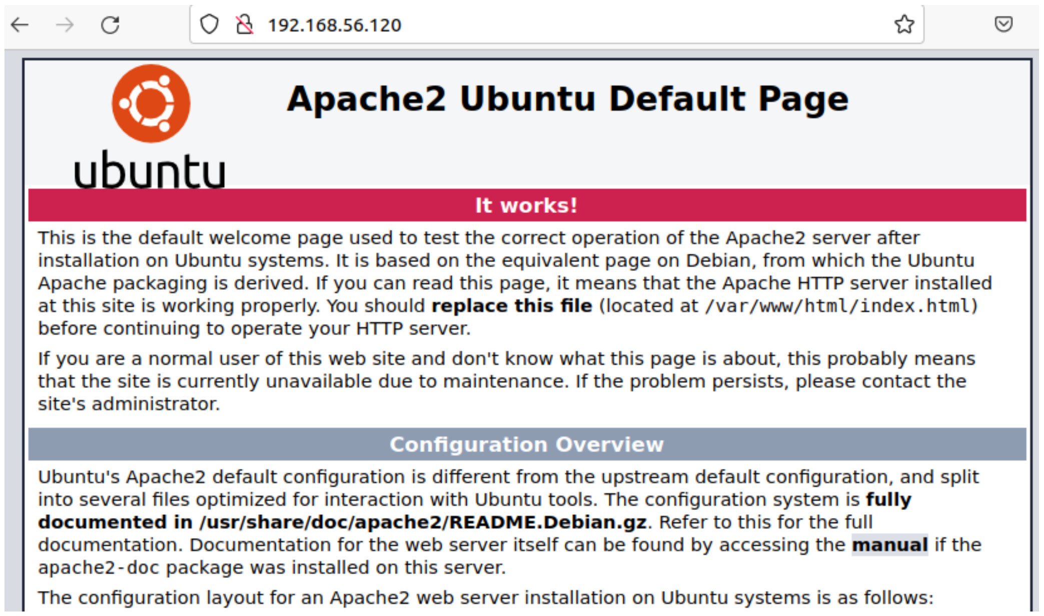
```
                    janssenvalencia@Workstation: ~/CPE232_VALENCIA
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                      install_apache.yml

---
- 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.



```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible-playbook --ask-become-pa
ss install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *******************************************************
*
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [install apache2 package] ***********************************************
*
changed: [192.168.56.102]
changed: [192.168.56.103]

PLAY RECAP *******************************************************************
*
192.168.56.102             : 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
```

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



192.168.56.120

Apache2 Ubuntu Default Page

ubuntu

It works!

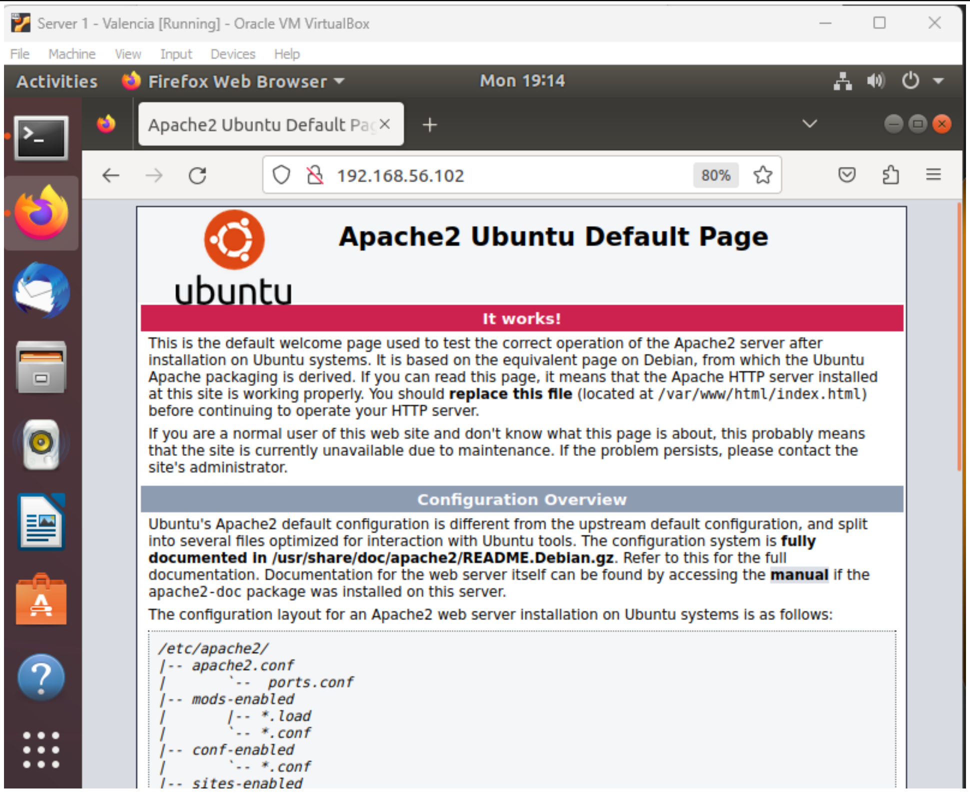
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at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

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

Configuration Overview

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

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


Server 1 - Valencia [Running] - Oracle VM VirtualBox

192.168.56.102

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.

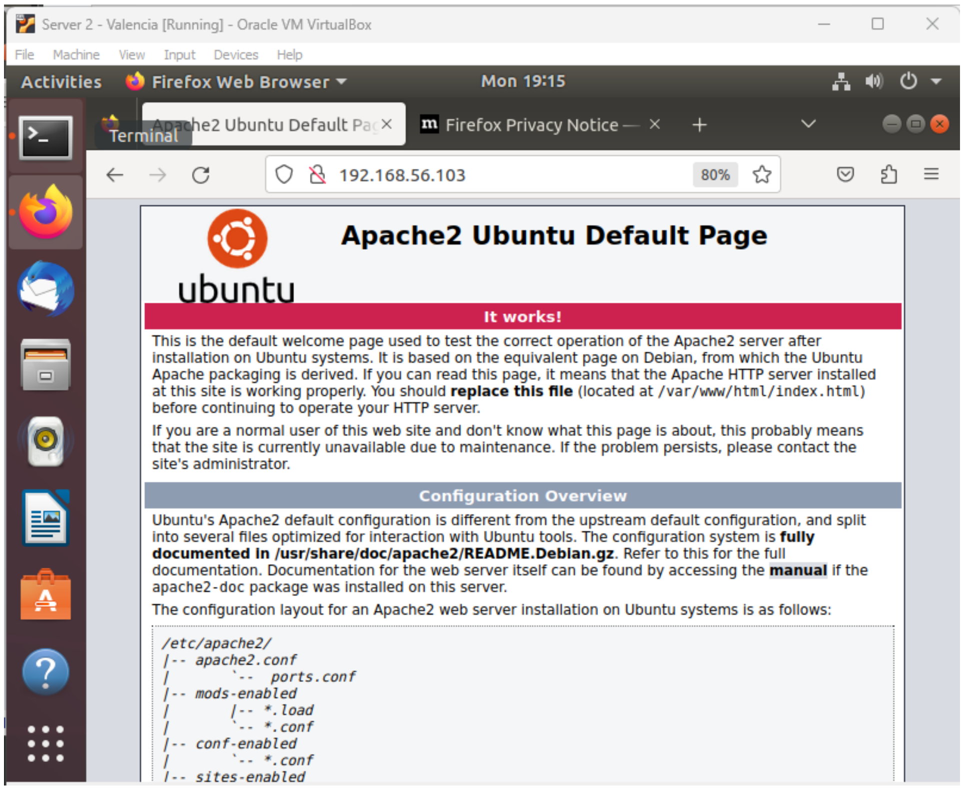
Configuration Overview

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
```

SERVER 1


Server 2 - Valencia [Running] - Oracle VM VirtualBox

192.168.56.103

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

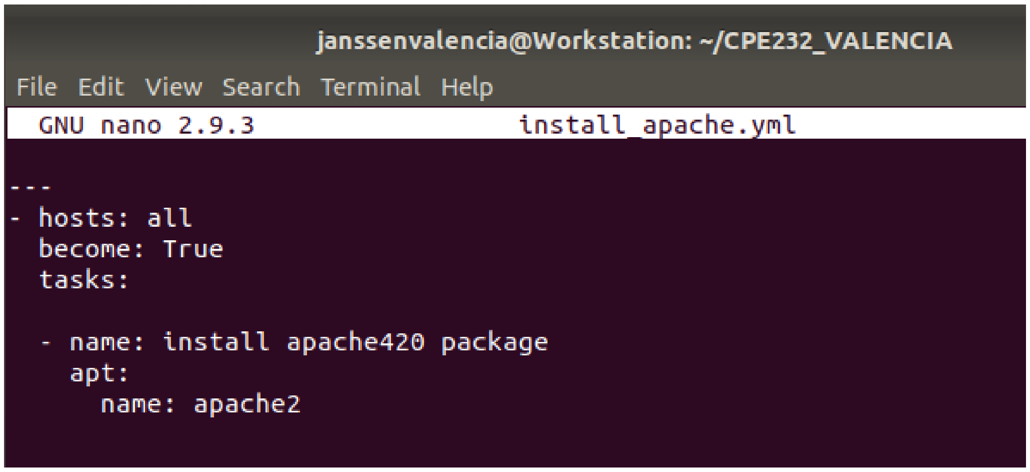
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation.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

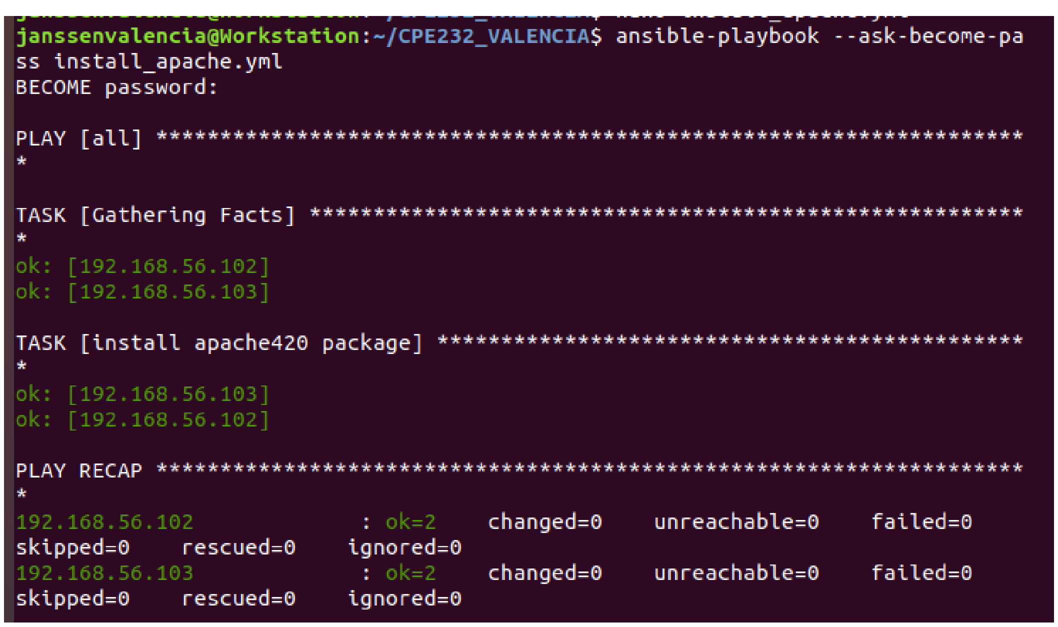
```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
```

SERVER 2

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? <u>**The same as the original command.**</u>

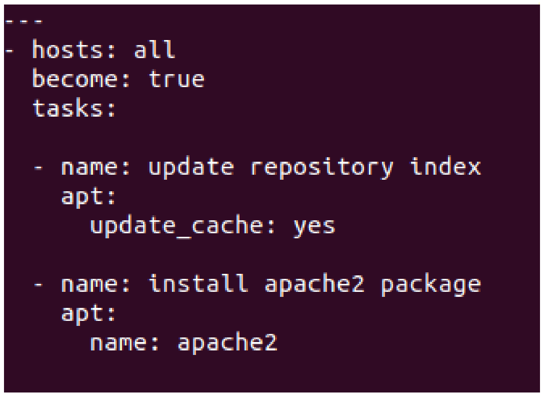

janssenvalencia@Workstation: ~/CPE232_VALENCIA
File Edit View Search Terminal Help

```
  GNU nano 2.9.3                      install_apache.yml

---
- hosts: all
  become: True
  tasks:

  - name: install apache420 package
    apt:
      name: apache2
```



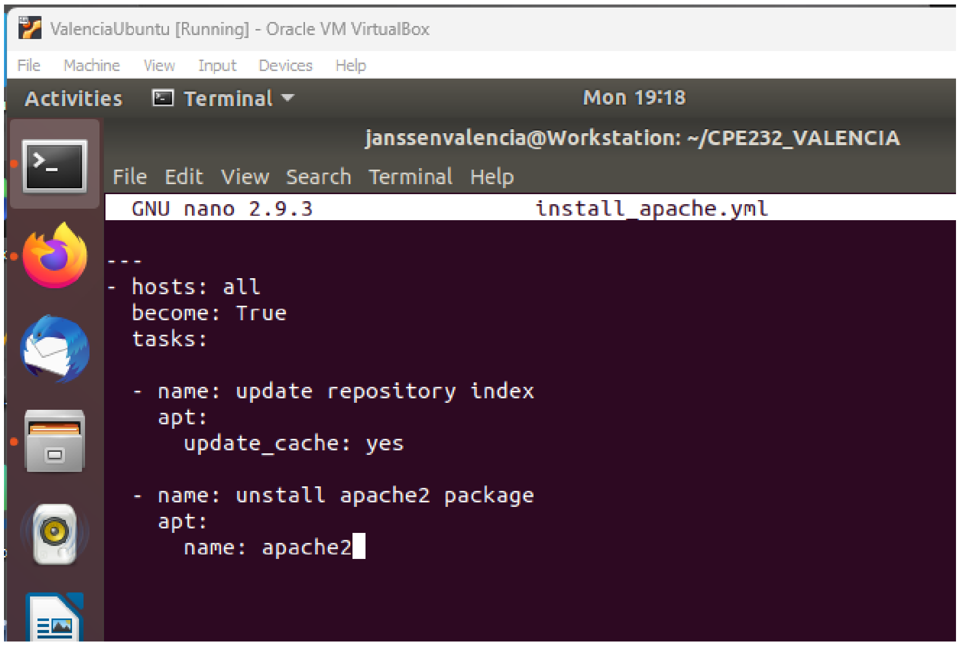
```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible-playbook --ask-become-pa
ss install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [install apache420 package] ***********************************************
*
ok: [192.168.56.103]
ok: [192.168.56.102]

PLAY RECAP *********************************************************************
*
192.168.56.102             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

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
```

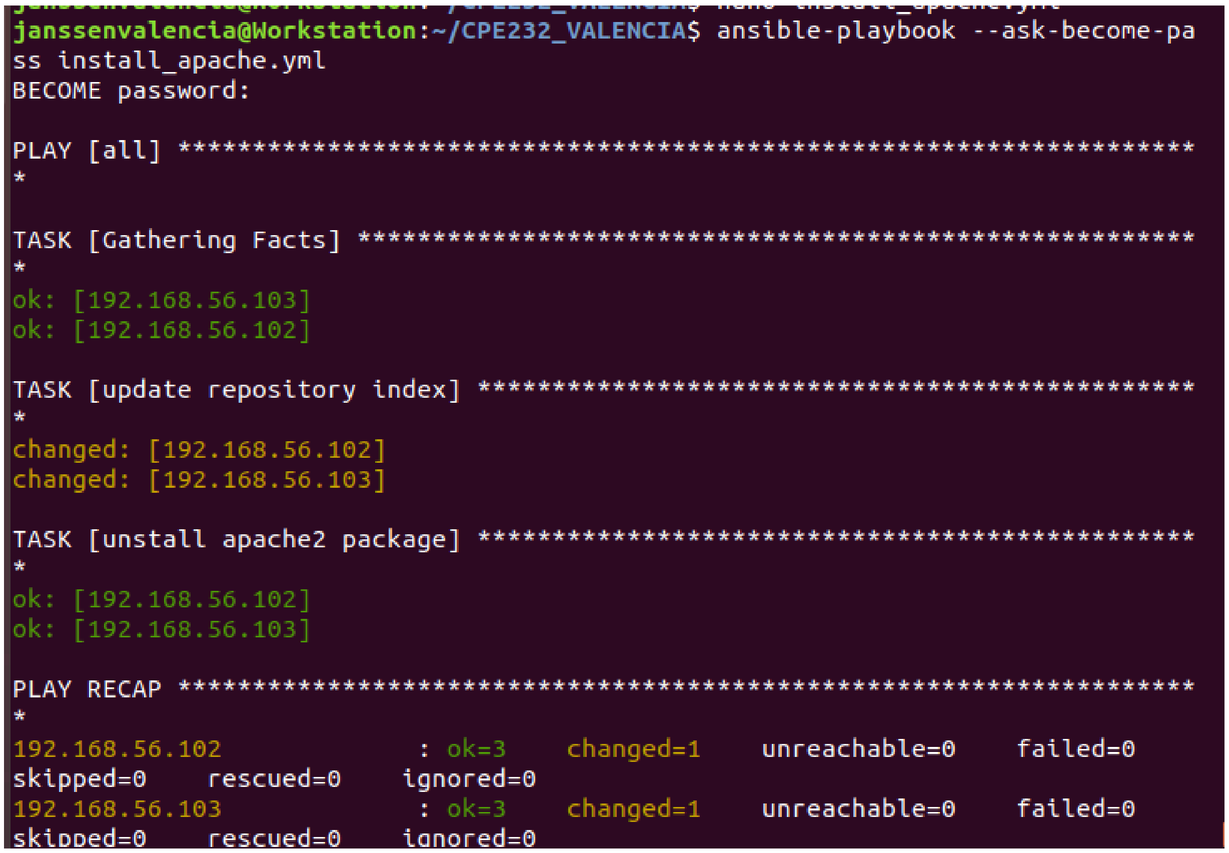
Save the changes to this file and exit.



ValenciaUbuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Mon 19:18
janssenvalencia@Workstation: ~/CPE232_VALENCIA
File Edit View Search Terminal Help

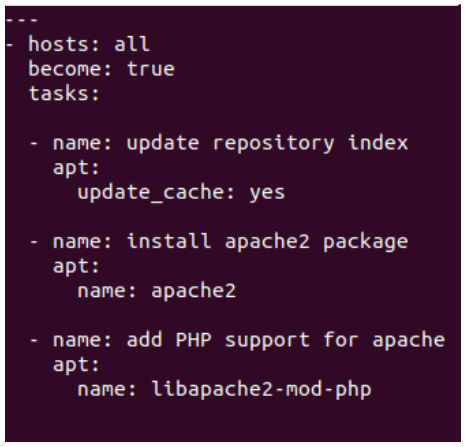
```
  GNU nano 2.9.3                  install_apache.yml

---
- hosts: all
  become: True
  tasks:

  - name: update repository index
    apt:
      update_cache: yes

  - name: unstall apache2 package
    apt:
      name: apache2
```

6. Run the playbook and describe the output. Did the new command change anything on the remote servers? **_Yes, Update repository index was changed, showing a total of 1 changes on each of the two servers._**



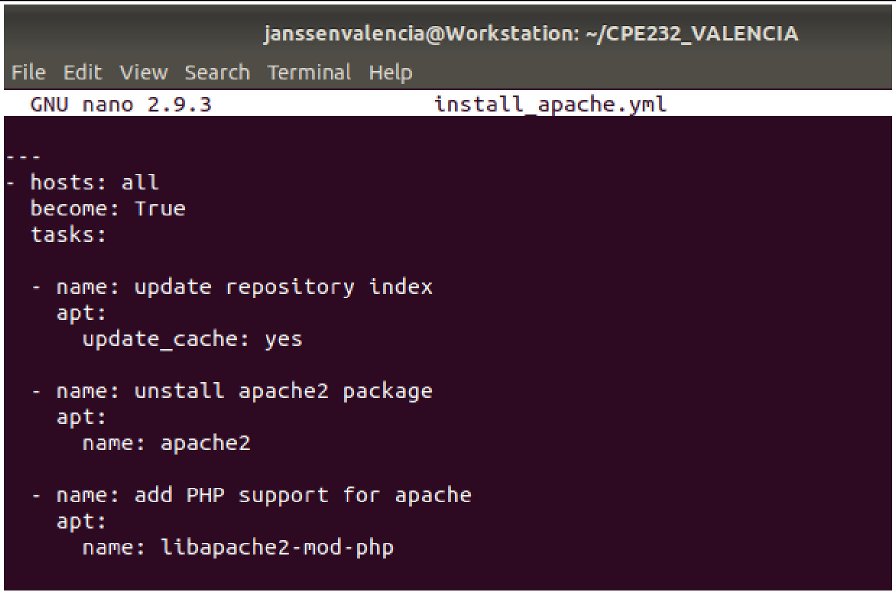
```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible-playbook --ask-become-pa
ss install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *******************************************************
*
ok: [192.168.56.103]
ok: [192.168.56.102]

TASK [update repository index] ***********************************************
*
changed: [192.168.56.102]
changed: [192.168.56.103]

TASK [unstall apache2 package] ***********************************************
*
ok: [192.168.56.102]
ok: [192.168.56.103]

PLAY RECAP *******************************************************************
*
192.168.56.102             : 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
```

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



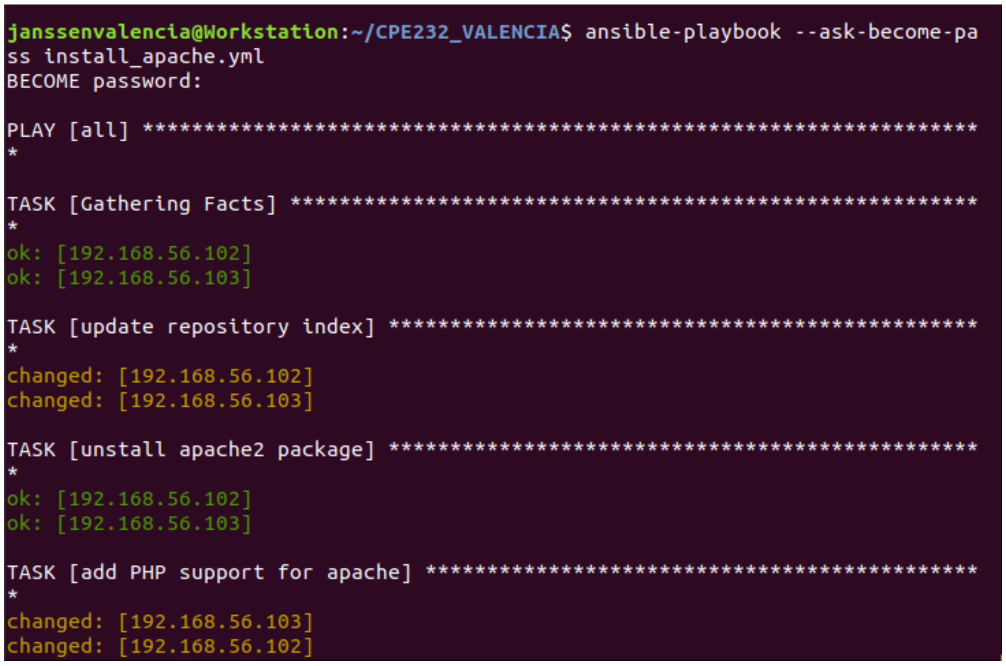
```
---
- hosts: all
  become: true
  tasks:

  - 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.

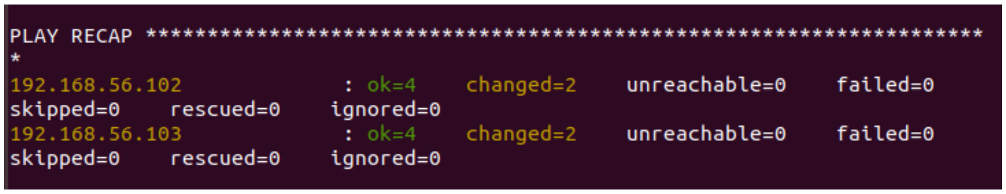

janssenvalencia@Workstation: ~/CPE232_VALENCIA
File Edit View Search Terminal Help

```
  GNU nano 2.9.3                    install_apache.yml

---
- hosts: all
  become: True
  tasks:

  - name: update repository index
    apt:
      update_cache: yes

  - name: unstall 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? **__Yes, update repository index is changed along with the added PHP support for apache, showing a total of 2 changes on each of the two servers.__**

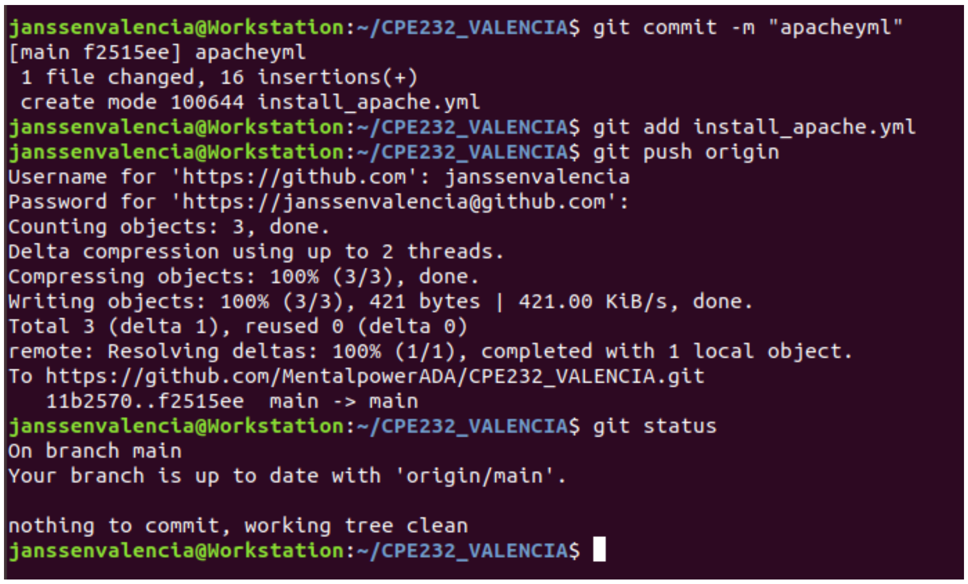


```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ ansible-playbook --ask-become-pa
ss install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *****************************************************
*
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [update repository index] *********************************************
*
changed: [192.168.56.102]
changed: [192.168.56.103]

TASK [unstall apache2 package] *********************************************
*
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [add PHP support for apache] ******************************************
*
changed: [192.168.56.103]
changed: [192.168.56.102]
```



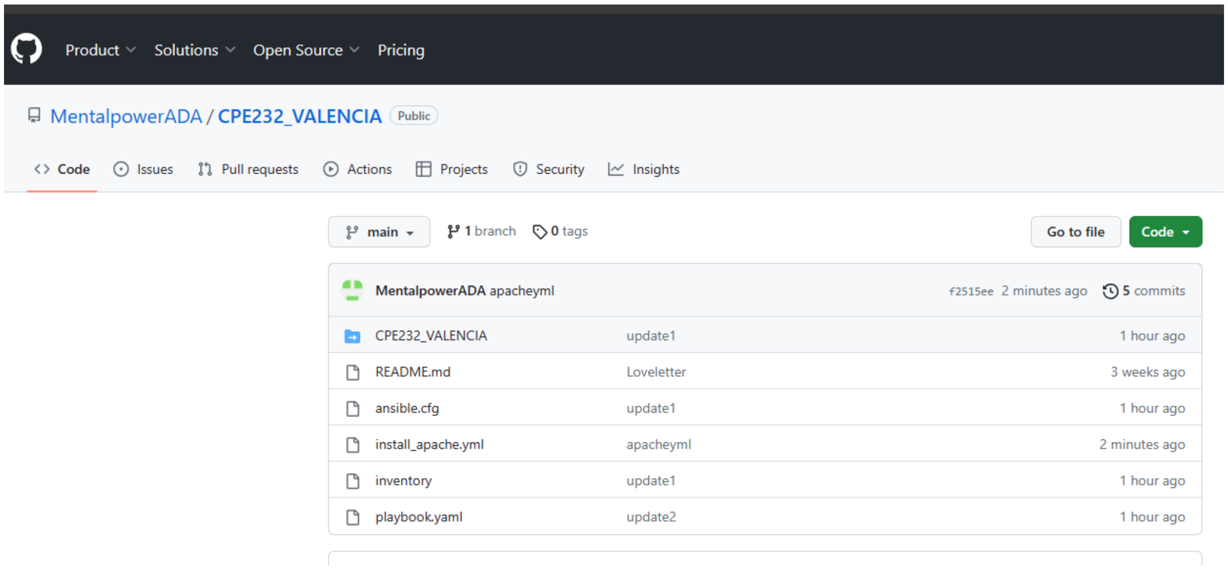
```
PLAY RECAP *****************************************************************
*
192.168.56.102             : ok=4    changed=2    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=4    changed=2    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0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

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



```
janssenvalencia@Workstation:~/CPE232_VALENCIA$ git commit -m "apacheyml"
[main f2515ee] apacheyml
 1 file changed, 16 insertions(+)
 create mode 100644 install_apache.yml
janssenvalencia@Workstation:~/CPE232_VALENCIA$ git add install_apache.yml
janssenvalencia@Workstation:~/CPE232_VALENCIA$ git push origin
Username for 'https://github.com': janssenvalencia
Password for 'https://janssenvalencia@github.com':
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 421 bytes | 421.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/MentalpowerADA/CPE232_VALENCIA.git
   11b2570..f2515ee  main -> main
janssenvalencia@Workstation:~/CPE232_VALENCIA$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
janssenvalencia@Workstation:~/CPE232_VALENCIA$
```

https://github.com/MentalpowerADA/CPE232_VALENCIA.git



MentalpowerADA apacheyml   f2515ee 2 minutes ago   5 commits

CPE232_VALENCIA	update1	1 hour ago
README.md	Loveletter	3 weeks ago
ansible.cfg	update1	1 hour ago
install_apache.yml	apacheyml	2 minutes ago
inventory	update1	1 hour ago
playbook.yaml	update2	1 hour ago

Reflections:

Answer the following:

1. **What is the importance of using a playbook?**
    - During the last semester we were tasked to automate numerous task using scripts The main difference between bash scripts to playbook is that playbooks improves consistency and reduces errors making it easier to troubleshoot different problems encountered thus also making it collaboration-friendly since it makes delegating task easier with the implementation of git.

2. **Summarize what we have done on this activity.**
    - Prior to doing this hands-on activity during the lecture we were linking our GitHub repositories to our local virtual machines using SSH and token keys, were installing Python3 and Ansible to the workstation, Prior to this also we were tasked to establish an SSH connection in between machines. On starting the hands-on activity itself we ran ad hoc commands such as updating and upgrading the repositories on our local machines remotely using SSH and Ansible. We were also making changes and viewing the status of our remote machines using Ansible. On writing our first playbook we created and modified our very first playbook, conducting experiments such as making changes to the script and seeing the results it gives. we automated the installation of Apache2 on both of the remote servers mainly SERVER1 and SERVER2 using automated playbooks. all of the performed activities are synced and linked to the GitHub repository.

Conclusion:

- All in all this hands-on activity has taught me how to work with ansible and GitHub repositories. I was able to implement commands that make changes to remote servers mainly the two virtual machines Server1 and Server2. I was able to implement and create my very first playbook that automates Ansible commands to the remote servers. These new skills and knowledge that I have learned in this hands-on activity will surely be fundamental in the upcoming activities and will be used by me somewhere in the future as an aspiring system administrator.