Name: Torrecampo, Juan Piolo S.	**Date Performed:** Sep 6, 2022
Course/Section: CPE 232 / CPE31S22	**Date Submitted:** Sep 6, 2022
Instructor: Dr. Jonathan Taylar	**Semester and SY:** 1st Sem, 2022 - 2023

Activity 4: Running Elevated Ad hoc Commands

1. **Objectives:**

1.1 Use commands that makes changes to remote machines

1.2 Use playbook in automating ansible commands

2. **Discussion:**

Provide screenshots for each task.

Elevated Ad hoc commands

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

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

Task 1: Run elevated ad hoc commands

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

   ansible all -m apt -a update_cache=true

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

   - **No, it display a bunch of errors.**

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

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.

```
piolo@workstation:~/ansible$ ansible all -m apt -a update_cache=true --become --
ask-become-pass
BECOME password:
192.168.56.101 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662427674,
    "cache_updated": true,
    "changed": true
}
192.168.56.103 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662427675,
    "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.

```
piolo@workstation:~/ansible$ ansible all -m apt -a name=vim-nox --become --ask-become-pass
BECOME password:
192.168.56.101 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662427674,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading state information...\nThe following additional p
a5.2-0 libruby3.0 rake ruby\n  ruby-net-telnet ruby-rubygems ruby-webrick ruby-xmlrpc ruby3.0\n  rubygems-integration\nSuggest
\nThe following NEW packages will be installed:\n  fonts-lato javascript-common libjs-jquery liblua5.2-0 libruby3.0 rake ruby\
-integration vim-nox\n0 upgraded, 14 newly installed, 0 to remove and 38 not upgraded.\nNeed to get 10.6 MB of archives.\nAfte
//ph.archive.ubuntu.com/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1 [2696 kB]\nGet:2 http://ph.archive.ubuntu.com/ubuntu ja
e.ubuntu.com/ubuntu jammy/main amd64 libjs-jquery all 3.6.0+dfsg+~3.5.13-1 [321 kB]\nIgn:3 http://ph.archive.ubuntu.com/ubuntu
ive.ubuntu.com/ubuntu jammy/universe amd64 liblua5.2-0 amd64 5.2.4-2 [125 kB]\nGet:5 http://ph.archive.ubuntu.com/ubuntu jammy
buntu.com/ubuntu jammy-updates/main amd64 ruby3.0 amd64 3.0.2-7ubuntu2.1 [50.1 kB]\nGet:7 http://ph.archive.ubuntu.com/ubuntu
ubuntu.com/ubuntu jammy/main amd64 ruby amd64 1:3.0~exp1 [5100 B]\nGet:9 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64
/main amd64 ruby-net-telnet all 0.1.1-2 [12.6 kB]\nGet:11 http://ph.archive.ubuntu.com/ubuntu jammy/universe amd64 ruby-webric
ates/main amd64 ruby-xmlrpc all 0.3.2-1ubuntu0.1 [24.9 kB]\nGet:13 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd6
u.com/ubuntu jammy-updates/main amd64 libruby3.0 amd64 3.0.2-7ubuntu2.1\nGet:14 http://ph.archive.ubuntu.com/ubuntu jammy/univ
ve.ubuntu.com/ubuntu jammy/main amd64 libjs-jquery all 3.6.0+dfsg+~3.5.13-1 [321 kB]\nGet:13 http://ph.archive.ubuntu.com/ubun
```

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, as we can see in the screenshot below when we issue the command "which vim" in different servers it displays as where the vim is installed.**

Server 1

```
piolo@server1:~/Desktop$ which vim
/usr/bin/vim
piolo@server1:~/Desktop$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/jammy,now 2:8.2.3995-1ubuntu2 amd64 [installed]
  Vi IMproved - enhanced vi editor - with scripting languages support

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

piolo@server1:~/Desktop$
```

Server 2

```
piolo@server2:~/Desktop$ which vim
/usr/bin/vim
piolo@server2:~/Desktop$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/jammy,now 2:8.2.3995-1ubuntu2 amd64 [installed]
  Vi IMproved - enhanced vi editor - with scripting languages support

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

piolo@server2:~/Desktop$
```

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

```
piolo@server1:~/Desktop$ cd /var/log
piolo@server1:/var/log$ ls
alternatives.log    dmesg.1.gz       kern.log.1
alternatives.log.1  dmesg.2.gz       lastlog
apt                 dmesg.3.gz       openvpn
auth.log            dmesg.4.gz       private
auth.log.1          dpkg.log         speech-dispatcher
boot.log            dpkg.log.1       syslog
boot.log.1          faillog          syslog.1
bootstrap.log       fontconfig.log   ubuntu-advantage.log
btmp                gdm3             ubuntu-advantage-timer.log
btmp.1              gpu-manager.log  ubuntu-advantage-timer.log.1
cups                hp               unattended-upgrades
dist-upgrade        installer        wtmp
dmesg               journal
dmesg.0             kern.log
piolo@server1:/var/log$ cd apt
piolo@server1:/var/log/apt$ cat history.log

Start-Date: 2022-09-06  09:42:38
Commandline: /usr/bin/apt-get -y -o Dpkg::Options::=--force-confdef -o Dpkg::Options::=--force-confol
d install vim-nox
Requested-By: piolo (1000)
Install: fonts-lato:amd64 (2.0-2.1, automatic), liblua5.2-0:amd64 (5.2.4-2, automatic), ruby-net-teln
et:amd64 (0.1.1-2, automatic), rubygems-integration:amd64 (1.18, automatic), libruby3.0:amd64 (3.0.2-
7ubuntu2.1, automatic), rake:amd64 (13.0.6-2, automatic), vim-nox:amd64 (2:8.2.3995-1ubuntu2), ruby:a
md64 (1:3.0~exp1, automatic), ruby3.0:amd64 (3.0.2-7ubuntu2.1, automatic), libjs-jquery:amd64 (3.6.0+
dfsg+~3.5.13-1, automatic), ruby-rubygems:amd64 (3.3.5-2, automatic), javascript-common:amd64 (11+nmu
1, automatic), ruby-xmlrpc:amd64 (0.3.2-1ubuntu0.1, automatic), ruby-webrick:amd64 (1.7.0-3, automati
c)
End-Date: 2022-09-06  09:42:58
piolo@server1:/var/log/apt$
```

Server 2

```
piolo@server2:~$ cd /var/log
piolo@server2:/var/log$ ls
alternatives.log    dmesg.1.gz       kern.log.1
alternatives.log.1  dmesg.2.gz       lastlog
apt                 dmesg.3.gz       openvpn
auth.log            dmesg.4.gz       private
auth.log.1          dpkg.log         speech-dispatcher
boot.log            dpkg.log.1       syslog
boot.log.1          faillog          syslog.1
bootstrap.log       fontconfig.log   ubuntu-advantage.log
btmp                gdm3             ubuntu-advantage-timer.log
btmp.1              gpu-manager.log  ubuntu-advantage-timer.log.1
cups                hp               unattended-upgrades
dist-upgrade        installer        wtmp
dmesg               journal
dmesg.0             kern.log
piolo@server2:/var/log$ cd apt
piolo@server2:/var/log/apt$ cat history.log

Start-Date: 2022-09-06  09:42:33
Commandline: /usr/bin/apt-get -y -o Dpkg::Options::=--force-confdef -o Dpkg::Options::=--force-confol
d install vim-nox
Requested-By: piolo (1000)
Install: fonts-lato:amd64 (2.0-2.1, automatic), liblua5.2-0:amd64 (5.2.4-2, automatic), ruby-net-teln
et:amd64 (0.1.1-2, automatic), rubygems-integration:amd64 (1.18, automatic), libruby3.0:amd64 (3.0.2-
7ubuntu2.1, automatic), rake:amd64 (13.0.6-2, automatic), vim-nox:amd64 (2:8.2.3995-1ubuntu2), ruby:a
md64 (1:3.0~exp1, automatic), ruby3.0:amd64 (3.0.2-7ubuntu2.1, automatic), libjs-jquery:amd64 (3.6.0+
dfsg+~3.5.13-1, automatic), ruby-rubygems:amd64 (3.3.5-2, automatic), javascript-common:amd64 (11+nmu
1, automatic), ruby-xmlrpc:amd64 (0.3.2-1ubuntu0.1, automatic), ruby-webrick:amd64 (1.7.0-3, automati
c)
End-Date: 2022-09-06  09:42:52
piolo@server2:/var/log/apt$
```

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

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

   - **The result command is the same when you are pinging the remote servers. Yes it is a success but it does not change anything to the remote servers. Because the "snapd" is already downloaded in the remote servers. To prove my analyzation, I provided a screenshot below of history logs of server 1 and server 2.ansi**

```
piolo@workstation:~/ansible$ ansible all -m apt -a name=snapd --become --ask-become-pass
BECOME password:
192.168.56.103 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662427675,
    "cache_updated": false,
    "changed": false
}
192.168.56.101 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662427674,
    "cache_updated": false,
    "changed": false
}
```

Server 1 - history.log

```
piolo@server1:/var/log/apt$ cat history.log

Start-Date: 2022-09-06  09:42:38
Commandline: /usr/bin/apt-get -y -o Dpkg::Options::=--force-confdef -o Dpkg::Options::=--force-confol
d install vim-nox
Requested-By: piolo (1000)
Install: fonts-lato:amd64 (2.0-2.1, automatic), liblua5.2-0:amd64 (5.2.4-2, automatic), ruby-net-teln
et:amd64 (0.1.1-2, automatic), rubygems-integration:amd64 (1.18, automatic), libruby3.0:amd64 (3.0.2-
7ubuntu2.1, automatic), rake:amd64 (13.0.6-2, automatic), vim-nox:amd64 (2:8.2.3995-1ubuntu2), ruby:a
md64 (1:3.0~exp1, automatic), ruby3.0:amd64 (3.0.2-7ubuntu2.1, automatic), libjs-jquery:amd64 (3.6.0+
dfsg+~3.5.13-1, automatic), ruby-rubygems:amd64 (3.3.5-2, automatic), javascript-common:amd64 (11+nmu
1, automatic), ruby-xmlrpc:amd64 (0.3.2-1ubuntu0.1, automatic), ruby-webrick:amd64 (1.7.0-3, automati
c)
End-Date: 2022-09-06  09:42:58
```

Server 2 - history.log

```
piolo@server2:/var/log/apt$ cat history.log

Start-Date: 2022-09-06  09:42:33
Commandline: /usr/bin/apt-get -y -o Dpkg::Options::=--force-confdef -o Dpkg::Options::=--force-confol
d install vim-nox
Requested-By: piolo (1000)
Install: fonts-lato:amd64 (2.0-2.1, automatic), liblua5.2-0:amd64 (5.2.4-2, automatic), ruby-net-teln
et:amd64 (0.1.1-2, automatic), rubygems-integration:amd64 (1.18, automatic), libruby3.0:amd64 (3.0.2-
7ubuntu2.1, automatic), rake:amd64 (13.0.6-2, automatic), vim-nox:amd64 (2:8.2.3995-1ubuntu2), ruby:a
md64 (1:3.0~exp1, automatic), ruby3.0:amd64 (3.0.2-7ubuntu2.1, automatic), libjs-jquery:amd64 (3.6.0+
dfsg+~3.5.13-1, automatic), ruby-rubygems:amd64 (3.3.5-2, automatic), javascript-common:amd64 (11+nmu
1, automatic), ruby-xmlrpc:amd64 (0.3.2-1ubuntu0.1, automatic), ruby-webrick:amd64 (1.7.0-3, automati
c)
End-Date: 2022-09-06  09:42:52
```

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.

- The “state=latest” is used when we used to install the latest or update

```
piolo@workstation:~/ansible$ ansible all -m apt -a "name=snapd state=latest" --become --ask-become-pass
BECOME password:
192.168.56.103 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662427675,
    "cache_updated": false,
    "changed": false
}
192.168.56.101 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1662427674,
    "cache_updated": false,
    "changed": false
}
```

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

Copying the ansible directory to the CPE232_piolo git folder.

```
piolo@workstation:~$ cp -r ansible CPE232_piolo
```

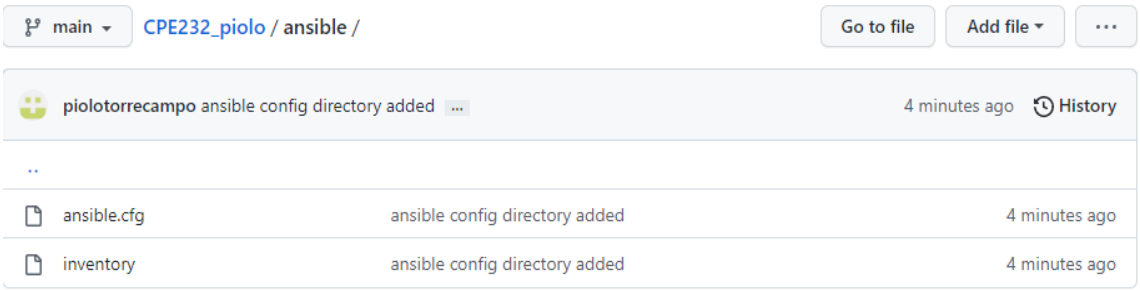
Listing the contents of CPE232_piolo

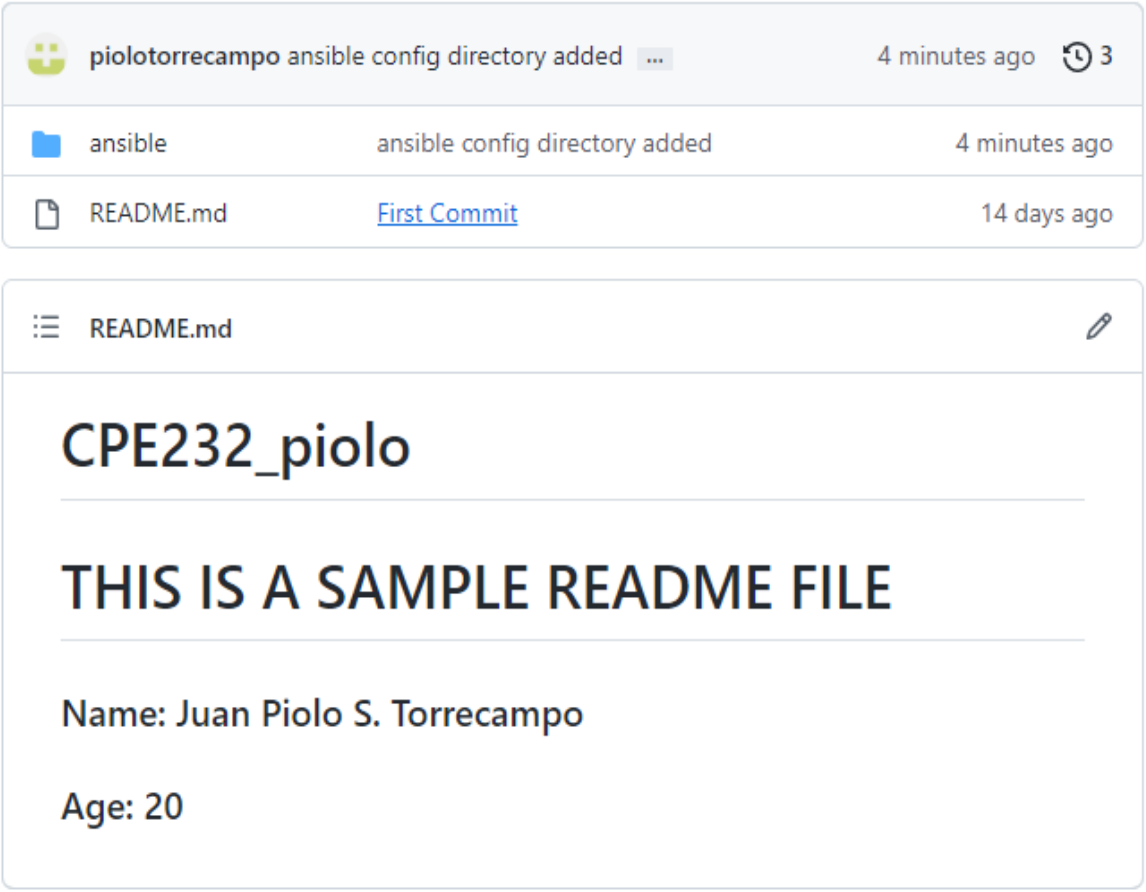
```
piolo@workstation:~/CPE232_piolo$ ll
total 20
drwxrwxr-x  4 piolo piolo 4096 Sep  6 10:54 ./
drwxr-x--- 18 piolo piolo 4096 Sep  6 10:54 ../
drwxrwxr-x  2 piolo piolo 4096 Sep  6 10:55 ansible/
drwxrwxr-x  8 piolo piolo 4096 Aug 23 09:42 .git/
-rw-rw-r--  1 piolo piolo   95 Aug 23 09:36 README.md
piolo@workstation:~/CPE232_piolo$
```

Committing the directory to Github

```
piolo@workstation:~/CPE232_piolo$ 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/

nothing added to commit but untracked files present (use "git add" to track)
piolo@workstation:~/CPE232_piolo$ git add *
piolo@workstation:~/CPE232_piolo$ git commit -m "ansible config directory added"
[main 8ac11be] ansible config directory added
 2 files changed, 13 insertions(+)
 create mode 100644 ansible/ansible.cfg
 create mode 100644 ansible/inventory
piolo@workstation:~/CPE232_piolo$ git push
The authenticity of host 'github.com (140.82.113.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 526 bytes | 526.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:piolotorrecampo/CPE232_piolo.git
   80c1419..8ac11be  main -> main
piolo@workstation:~/CPE232_piolo$
```

Github Repository





Task 2: Writing our First Playbook

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

   When the editor appears, type the following:

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

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

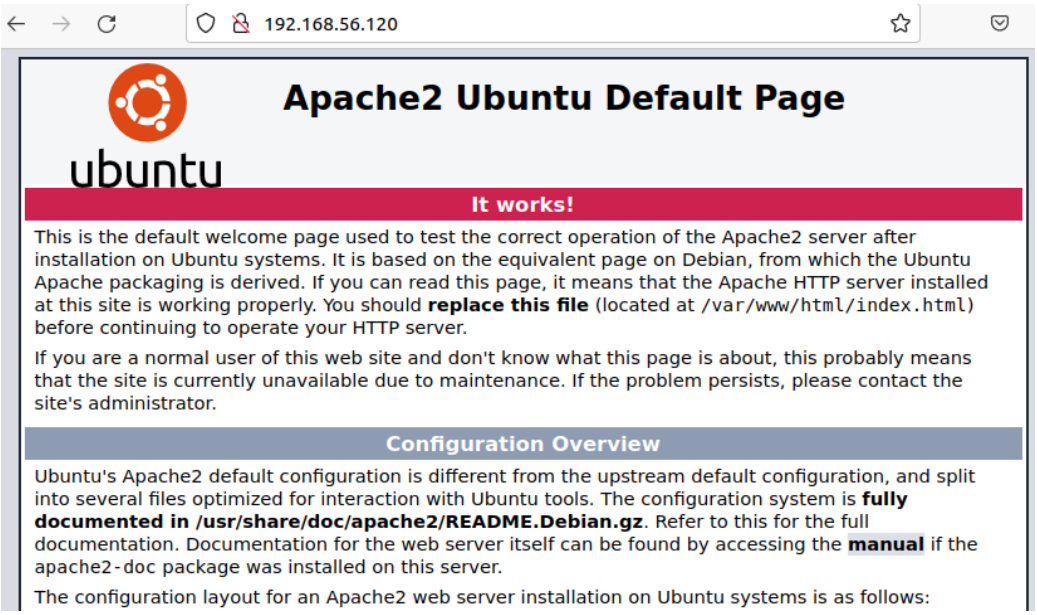
Make sure to save the file. Take note also of the alignments of the texts.

```
  GNU nano 6.2                    install_apache.yml
---
- hosts: all
  become: true
  tasks:
  - name: install apache2 package
    apt:
      name: apache2
```

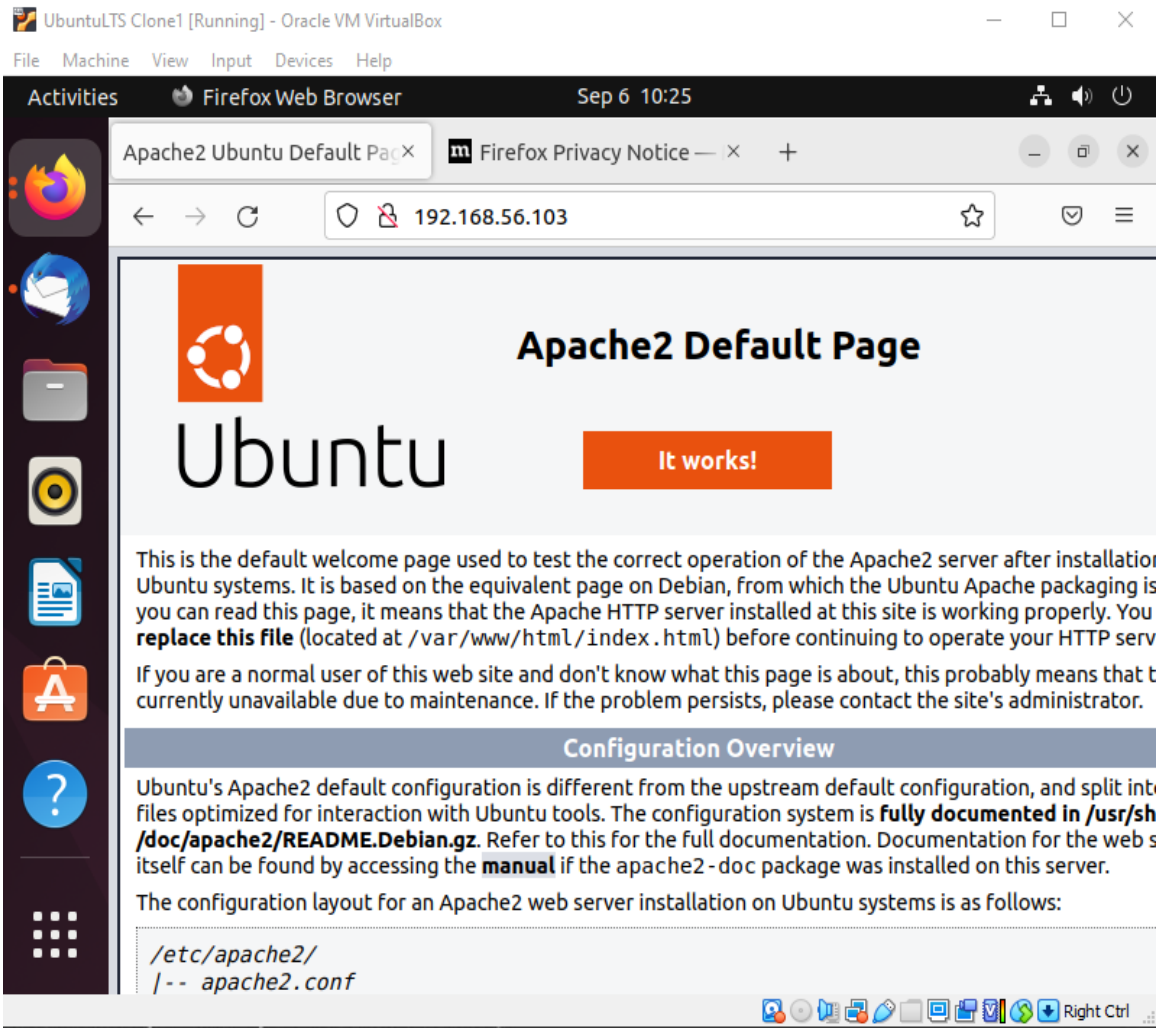
2. Run the yml file using the command: *ansible-playbook --ask-become-pass install_apache.yml*. Describe the result of this command.

```
piolo@workstation:~/ansible$ ansible-playbook --ask-become-pass install_apache.y
ml
BECOME password:

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

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

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

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



Server 1


UbuntuLTS Clone1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Firefox Web Browser
Sep 6 10:25
Apache2 Ubuntu Default Pa
Firefox Privacy Notice —
192.168.56.103
Apache2 Default Page
Ubuntu
It works!
This is the default welcome page used to test the correct operation of the Apache2 server after installatio
Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is
you can read this page, it means that the Apache HTTP server installed at this site is working properly. You
replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP serv
If you are a normal user of this web site and don't know what this page is about, this probably means that t
currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.
Configuration Overview
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split int
files optimized for interaction with Ubuntu tools. The configuration system is fully documented in /usr/sh
/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web s
itself can be found by accessing the manual if the apache2-doc package was installed on this server.
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:
/etc/apache2/
|-- apache2.conf
Right Ctrl

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?

Edited "install_apache.yml" Package Name

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

  - name: install nano package
    apt:
      name: nano
```

Ansible Output

```
piolo@workstation:~/ansible$ ansible-playbook --ask-become-pass install_apache.y
ml
BECOME password:

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

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

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

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

- **Since the nano is installed to the remote server, it shows ok only because the software is in the latest and installed to the remote servers.**

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

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

  - name: update repository index
    apt:
      update_cache: yes

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

Save the changes to this file and exit.

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

  - name: update repository index
    apt:
      update_cache: yes

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

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

```
piolo@workstation:~/ansible$ ansible-playbook --ask-become-pass install_apache.y
ml
BECOME password:

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

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

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

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

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

- **Yes, according to the "PLAY RECAP" there are 1 change in both remote servers. This is because we issue the update_cache which updates its repository.**

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

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

  - name: update repository index
    apt:
      update_cache: yes

  - name: install apache2 package
    apt:
      name: apache2

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

Save the changes to this file and exit.

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

  - name: update repository index
    apt:
      update_cache: yes

  - name: install apache2 package
    apt:
      name: apache2

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

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

```
piolo@workstation:~/ansible$ ansible-playbook --ask-become-pass install_apache.y
ml
BECOME password:

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

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

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

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

TASK [add PHP support for apache] **********************************************
changed: [192.168.56.101]
changed: [192.168.56.103]

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

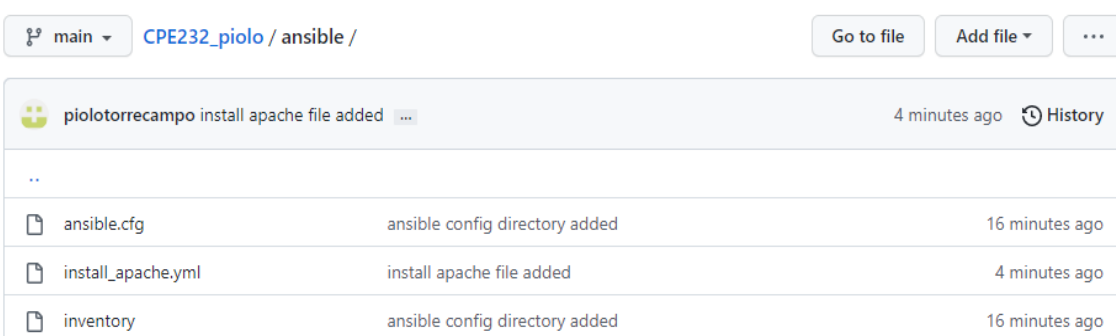
- **Yes, since the "libapache2-mod-php" is not installed in the system then the ansible will install this application in the 2 remote servers. Also, it will update the cache again because it is stated in the yml file.**

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

Adding the install_apache.yml file in the Github Repository

```
piolo@workstation:~/CPE232_piolo$ mv install_apache.yml ansible
piolo@workstation:~/CPE232_piolo$ ll
total 20
drwxrwxr-x  4 piolo piolo 4096 Sep  6 11:07 ./
drwxr-x--- 18 piolo piolo 4096 Sep  6 10:54 ../
drwxrwxr-x  2 piolo piolo 4096 Sep  6 11:07 ansible/
drwxrwxr-x  8 piolo piolo 4096 Sep  6 11:06 .git/
-rw-rw-r--  1 piolo piolo   95 Aug 23 09:36 README.md
piolo@workstation:~/CPE232_piolo$ git add *
piolo@workstation:~/CPE232_piolo$ 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/install_apache.yml

piolo@workstation:~/CPE232_piolo$ git commit -m "install apache file added"
[main e00aa2d] install apache file added
 1 file changed, 16 insertions(+)
 create mode 100644 ansible/install_apache.yml
piolo@workstation:~/CPE232_piolo$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 552 bytes | 552.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:piolotorrecampo/CPE232_piolo.git
   8ac11be..e00aa2d  main -> main
piolo@workstation:~/CPE232_piolo$
```

Github CPE232_piolo Repository



Github Repository Link: *https://github.com/piolotorrecampo/CPE232_piolo.git*

Reflections:

Answer the following:

1. What is the importance of using a playbook?

- The importance of using playbook is that when we have different actions that we want to perform in the different machines simontaunely, we can issue a yml and run in ansible. This ansible feature is useful in performing different configurations in multiple servers. Also, it reduces the time for system administrators in configuring different machines.

2. Summarize what we have done on this activity.

- To sum up this activity, the first part is focused on performing elevated and ad hoc commands using ansible. It uses the "–become –ask-become-pass" to elevate the privilege in performing commands in remote servers like initiating an installation of vim as I performed in task one. The task one provides the verification in the two servers, if there is a change in the system the actions are provided in the file of /etc/apt/library.log. On the other hand, task two introduces me to the yml configuration of ansible playbook. This file is a series of actions that executes when it is called by the ansible software. Also, it includes the use of reading the "PLAY RECAP" to determine if there are any changes to the remote servers.

Honor Pledge:

"I affirm that I will not give or receive unauthorized help on this activity and that all will be my own."

\