

Name: Victor Ortega	Date Performed: 09/12/23
Course/Section:	Date Submitted: 09/12/23
Instructor: Engr. Roman Richard	Semester and SY: 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: <i>Provide screenshots for each task.</i> 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 <i>sudo apt update</i> when we want to download package information from all configured resources. The sources often defined in <i>/etc/apt/sources.list</i> file and other files located in <i>/etc/apt/sources.list.d/</i> 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 cessful?	

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.

ansible all -m apt -a update_cache=true

```
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
victor@workstation:~$ ansible all -m apt -a update_cache=true
127.0.0.1 | 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)"
}
victor@workstation:~$
```

ansible all -m apt -a update_cache=true --become --ask-become-pass

```
victor@workstation:~$ ansible all -m apt -a update_cache=true --become --ask-become-pass
BECOME password:
127.0.0.1 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "cache_update_time": 1694511099,
  "cache_updated": true,
  "changed": true
}
victor@workstation:~$
```

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.

```
Victor@workstation:~$ ansible all -m apt -a name=vim-nox --become --ask-become-pass
BECOME password:
127.0.0.1 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "cache_update_time": 1694511757,
  "cache_updated": false,
  "changed": true,
  "stderr": "",
  "stderr_lines": [],
  "stdout": "Reading package lists...\nBuilding dependency tree...\nReading state information...\nThe following additional package
s will be installed:\n fonts-lato javascript-common libjs-jquery liblua5.2-0 libruby3.0 rake ruby\n ruby-net-telnet ruby-rubygems
ruby-webrick ruby-xmllrpc ruby3.0\n rubygems-integration vim-runtime\nSuggested packages:\n apache2 | lighttpd | httpd ri ruby-dev
bundler cscope vim-doc\nThe following NEW packages will be installed:\n fonts-lato javascript-common libjs-jquery liblua5.2-0 libr
uby3.0 rake ruby\n ruby-net-telnet ruby-rubygems ruby-webrick ruby-xmllrpc ruby3.0\n rubygems-integration vim-nox vim-runtime\n0 upg
raded, 15 newly installed, 0 to remove and 2 not upgraded.\nNeed to get 17.5 MB of archives.\nAfter this operation, 76.4 MB of addit
ional disk space will be used.\nGet:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1 [2696 kB]\nGet:2 h
ttp://ph.archive.ubuntu.com/ubuntu jammy/main amd64 javascript-common all 11+nmu1 [5936 B]\nGet:3 http://ph.archive.ubuntu.com/ubunt
u jammy/main amd64 libjs-jquery all 3.6.0+dfsg+~3.5.13-1 [321 kB]\nGet:4 http://ph.archive.ubuntu.com/ubuntu jammy/universe amd64 li
blua5.2-0 amd64 5.2.4-2 [125 kB]\nGet:5 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 rubygems-integration all 1.18 [5336 B]\
nGet:6 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby3.0 amd64 3.0.2-7ubuntu2.4 [50.1 kB]\nGet:7 http://ph.archiv
e.ubuntu.com/ubuntu jammy/main amd64 ruby-rubygems all 3.3.5-2 [228 kB]\nGet:8 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64
ruby amd64 1:3.0-expi [5100 B]\nGet:9 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 rake all 13.0.6-2 [61.7 kB]\nGet:10 http:
//ph.archive.ubuntu.com/ubuntu jammy/main amd64 ruby-net-telnet all 0.1.1-2 [12.6 kB]\nGet:11 http://ph.archive.ubuntu.com/ubuntu ja
mmy/universe amd64 ruby-webrick all 1.7.0-3 [51.8 kB]\nGet:12 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby-xmllr
pc all 0.3.2-1ubuntu0.1 [24.9 kB]\nGet:13 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libruby3.0 amd64 3.0.2-7ubunt
u2.4 [5113 kB]\nGet:14 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 vim-runtime all 2:8.2.3995-1ubuntu2.11 [6828 kB]
\nGet:15 http://ph.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 vim-nox amd64 2:8.2.3995-1ubuntu2.11 [1941 kB]\nFetched 17
.5 MB in 7s (2371 kB/s)\nSelecting previously unselected package fonts-lato.\n(Reading database ... \n(Reading database ... 5%\r(R
eading database ... 10%\r(Reading database ... 15%\r(Reading database ... 20%\r(Reading database ... 25%\r(Reading database ... 30%\r
(Reading database ... 35%\r(Reading database ... 40%\r(Reading database ... 45%\r(Reading database ... 50%\r(Reading database ... 5
5%\r(Reading database ... 60%\r(Reading database ... 65%\r(Reading database ... 70%\r(Reading database ... 75%\r(Reading database ...
80%\r(Reading database ... 85%\r(Reading database ... 90%\r(Reading database ... 95%\r(Reading database ... 100%\r(Reading databas
```

- 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?

```
Victor@workstation:~$ which vim && apt search vim-nox
/usr/bin/vim
Sorting... Done
Full Text Search... Done
vim-nox/jammy-updates,jammy-security,now 2:8.2.3995-1ubuntu2.11 amd64 [installed]
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
```

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`.



```
GNU nano 6.2 history.log
Start-Date: 2023-08-07 22:53:16
Commandline: apt-get --yes -oDebug::pkgDepCache::AutoInstall=yes --force=yes upgrade
Upgrade: dpkg:amd64 (1.21.1ubuntu2, 1.21.1ubuntu2.2), libxtables12:amd64 (1.8.7-1ubuntu5, 1.8.7-1ubuntu5.1), networkd-dispatcher:amd64 (1.21-1ubuntu1, 1.21-1ubuntu1.1)
End-Date: 2023-08-07 22:53:29

Start-Date: 2023-08-07 22:53:30
Commandline: apt-get --yes -oDebug::pkgDepCache::AutoInstall=yes --force=yes dist-upgrade
Install: systemd-hwe-hwdb:amd64 (249.11.3, automatic)
Upgrade: udev:amd64 (249.11-0ubuntu3, 249.11-0ubuntu3.9), libudev1:amd64 (249.11-0ubuntu3, 249.11-0ubuntu3.9)
End-Date: 2023-08-07 22:53:31

Start-Date: 2023-08-07 22:53:38
Commandline: apt-get --yes -oDebug::pkgDepCache::AutoInstall=yes install linux-generic-hwe-22.04 adduser base-passwd bash bsdutils
Install: kerneloops:amd64 (0.12+git20140509-6ubuntu5), openvpn:amd64 (2.5.5-1ubuntu3.1), fontconfig:amd64 (2.13.1-4.2ubuntu5), libvncserver:amd64 (0.9.11-1ubuntu1)
End-Date: 2023-08-07 22:55:51

Start-Date: 2023-08-07 22:55:55
Commandline: apt-get --yes -oDebug::pkgDepCache::AutoInstall=yes install casper fonts-arphic-ukai fonts-arphic-uming gnome-user-docs
Install: libreoffice-l10n-en-gb:amd64 (1:7.3.7-0ubuntu0.22.04.3), libm17n-0:amd64 (1.8.0-4), libreoffice-l10n-en-za:amd64 (1:7.3.7-0ubuntu0.22.04.3)
End-Date: 2023-08-07 22:57:57

Start-Date: 2023-09-12 13:27:48
Requested-By: ubuntu (999)
End-Date: 2023-09-12 13:27:48

Start-Date: 2023-09-12 13:28:05
Requested-By: ubuntu (999)
Purge: libreoffice-l10n-en-gb:amd64 (1:7.3.7-0ubuntu0.22.04.3), libm17n-0:amd64 (1.8.0-4), libreoffice-l10n-en-za:amd64 (1:7.3.7-0ubuntu0.22.04.3)
End-Date: 2023-09-12 13:29:28

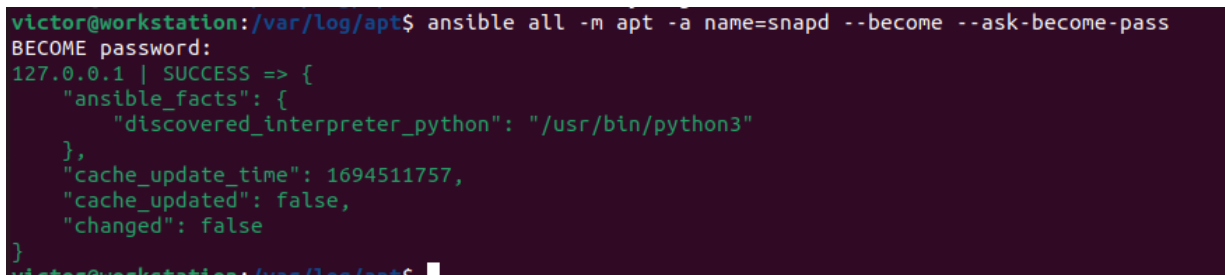
Start-Date: 2023-09-12 13:29:29
Requested-By: ubuntu (999)
Purge: libdhash1:amd64 (0.6.2-1), libini-config5:amd64 (0.6.2-1), libcollection4:amd64 (0.6.2-1), libc-ares2:amd64 (1.18.1-1ubuntu0.22.04.3)
End-Date: 2023-09-12 13:29:32

[ Read 82 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo      M-G Copy
```

3. This time, we will install a package called `snapt`. 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=snapt --become --ask-become-pass`

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



```
victor@workstation:/var/log/apt$ ansible all -m apt -a name=snapt --become --ask-become-pass
BECOME password:
127.0.0.1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "cache_update_time": 1694511757,
  "cache_updated": false,
  "changed": false
}
victor@workstation:/var/log/apt$
```

Answer: In this case, the command will install the `snapt` package on all remote hosts. The `snapt` package is a daemon that allows users to install and manage snap packages, which are a type of lightweight package that can be installed on Linux systems.

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

```
victor@workstation:/var/log/apt$ ansible all -m apt -a "name=snapd state=latest" --become --ask-become-pass
BECOME password:
127.0.0.1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "cache_update_time": 1694511757,
  "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.

```
victor@workstation:~/CPE232_Ortega$ ls
Ansible  README.md
victor@workstation:~/CPE232_Ortega$ git add .
victor@workstation:~/CPE232_Ortega$ git commit -m "Update" Update
error: pathspec 'Update' did not match any file(s) known to git
victor@workstation:~/CPE232_Ortega$ git commit -m "Update"
[main 1426abb] Update
 1 file changed, 2 insertions(+)
 create mode 100644 Ansible
victor@workstation:~/CPE232_Ortega$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 285 bytes | 285.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:qvbTor/CPE232_Ortega.git
 66ce615..1426abb  main -> main
victor@workstation:~/CPE232_Ortega$
```

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
--
- 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.

```
victor@workstation:~$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

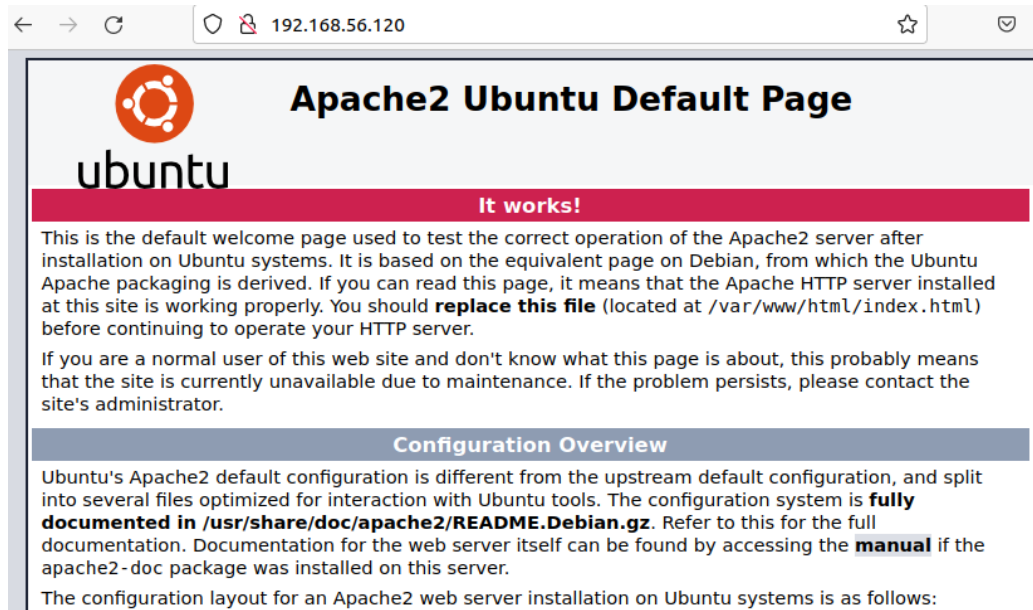
PLAY [all] *****

TASK [Gathering Facts] *****
ok: [127.0.0.1]

TASK [install apache2 package] *****
changed: [127.0.0.1]

PLAY RECAP *****
127.0.0.1                : 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.





Apache2 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:

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

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?
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
---
- 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?

```
victor@workstation:~$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [127.0.0.1]

TASK [update repository index] *****
changed: [127.0.0.1]

TASK [install apache2 package] *****
ok: [127.0.0.1]

PLAY RECAP *****
127.0.0.1 : 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.

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

```
victor@workstation:~$ nano install_apache.yml
victor@workstation:~$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [127.0.0.1]

TASK [update repository index] *****
changed: [127.0.0.1]


TASK [install apache2 package] *****
ok: [127.0.0.1]

TASK [add PHP support for apache] *****
changed: [127.0.0.1]

PLAY RECAP *****
127.0.0.1 : 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.

```
victor@workstation:~/CPE232_Ortega$ git add install_apache.yml
victor@workstation:~/CPE232_Ortega$ git commit -m "Apache Added"
[main 70689d3] Apache Added
 1 file changed, 16 insertions(+)
 create mode 100644 install_apache.yml
victor@workstation:~/CPE232_Ortega$ 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), 456 bytes | 456.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:qvbTor/CPE232_Ortega.git
 1426abb..70689d3  main -> main
```

 qvbTor / CPE232_Ortega


Type to search




[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)


 **CPE232_Ortega** Public

[Pin](#) [Unwatch](#) 1 [Fork](#) 0

[main](#) [1 branch](#) [0 tags](#) [Go to file](#) [Add file](#) [Code](#)

 **qvbTor** Apache Added 70689d3 1 minute ago 4 commits

 Ansible	Update	32 minutes ago
 README.md	Beep	2 weeks ago
 install_apache.yml	Apache Added	1 minute ago

README.md 

SYSAD

No description, website, or provided.

[Readme](#)

[Activity](#)

[0 stars](#)

[1 watching](#)

[0 forks](#)

Releases

No releases published

[Create a new release](#)

```
main
qvbTor committed 1 minute ago 1 parent 1426abb commit 70689d3

Showing 1 changed file with 16 additions and 0 deletions.

16 install_apache.yml
... -0,0 +1,16 ...

1 + ---
2 + - hosts: all
3 +   become: true
4 +   tasks:
5 +
6 +     - name: update repository index
7 +       apt:
8 +         update_cache: yes
9 +
10 +    - name: install apache2 package
11 +      apt:
12 +        name: apache2
13 +
14 +    - name: add PHP support for apache
15 +      apt:
16 +        name: libapache2-mod-php
```

0 comments on commit 70689d3 Lock conversation

Reflections:

Answer the following:

1. What is the importance of using a playbook?

Answer: A playbook is a document that describes the steps involved in carrying out a specific task or process. It can be used in a variety of settings, such as businesses, big companies, etc.

2. Summarize what we have done on this activity.

Answer: Ansible playbooks are used to automate tasks on remote systems. They are written in a human-friendly language and can be used to perform a variety of tasks, such as installing software, configuring systems, and deploying applications. The overall activity involves installing Ansible and configuring it by adding the local host, which is 127.0.0.1 or a specific address. Then, the playbook setup will be done so that both server 1 and 2 can access it because they have the same local host.

