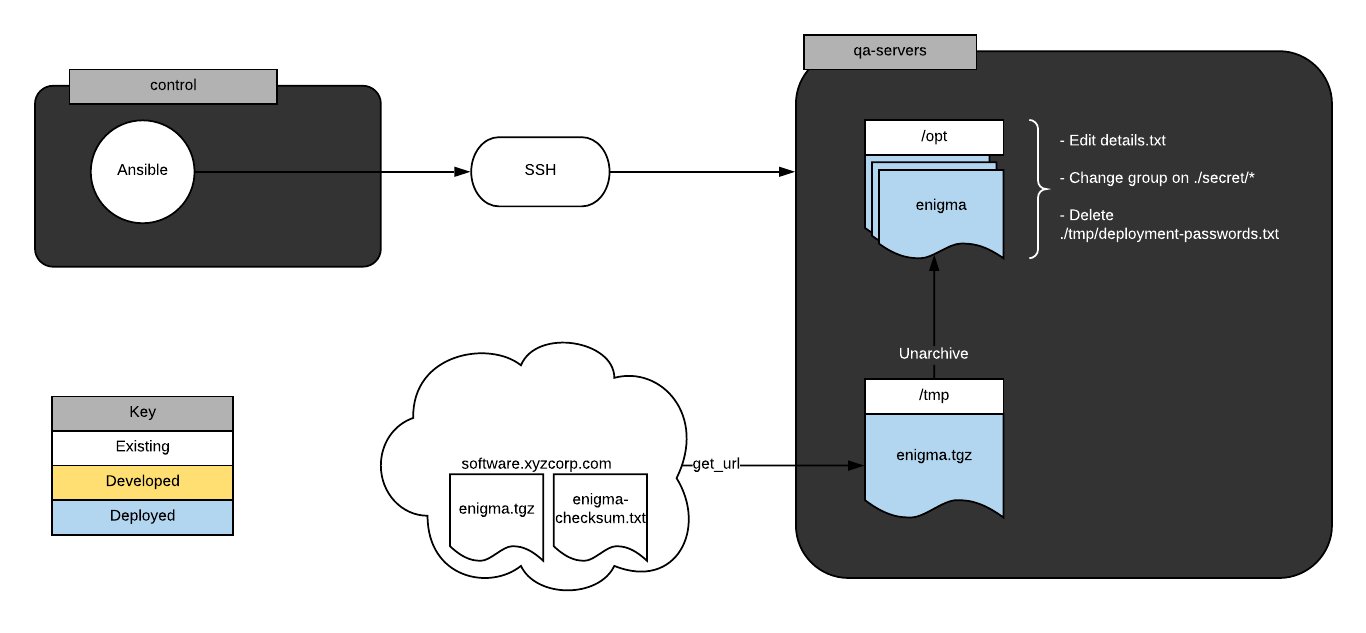
# Ansible Practitioner 8- 18.

8

## File Manipulation with Ansible.



# File Manipulation with Ansible

## Introduction

## A common theme in everyday systems administration is the need to work with system files. It follows that any good configuration management or automation engine must be able to do the same. This exercise challenges students to use Ansible to execute varying tasks on remote systems involving file manipulation.

## Solution

Log in to the Ansible Control Node lab server using the credentials provided:

ssh cloud\_user@<PUBLIC\_IP\_ADDRESS>

Become the ansible user:

su ansible

NOTE: All commands throughout the lab will be run from the Ansible Control Node lab server.

### Download a file to /tmp on each host in qa-servers and verify the sha256 checksum

Get the checksum value:

curl http://software.xyzcorp.com/enigma-checksum.txt

Copy the checksum value in the output, and paste it into a plaintext file (you'll need it for the next command).

Download a file to /tmp/ on each host in qa-servers, replacing <CHECKSUM\_VALUE> with the value you just copied:

ansible qa-servers -m get\_url -a "url=http://software.xyzcorp.com/enigma.tgz dest=/tmp/enigma.tgz checksum=sha256:<CHECKSUM\_VALUE>"

### Extract /tmp/enigma.tgz to /opt/ on all hosts in qa-servers

Run the command:

ansible qa-servers -b -m unarchive -a "src=/tmp/enigma.tgz dest=/opt/ remote\_src=yes"

### Update a line of text on each server in qa-servers

Update the line of text "DEPLOY\_CODE" to "CODE\_RED" in /opt/enigma/details.txt on each server in qa-servers:

ansible qa-servers -b -m lineinfile -a "regexp=DEPLOY\_CODE line=CODE\_RED path=/opt/enigma/details.txt"

### Set the group ownership of a directory on each host in qa-servers

Set the group ownership of the directory /opt/enigma/secret/ and each file contained within the directory so that the group owner is protected for each host in qa-servers:

ansible qa-servers -b -m file -a "recurse=yes state=directory path=/opt/enigma/secret group=protected"

### Delete a file on all servers

Delete the file /opt/enigma/tmp/deployment-passwords.txt on all servers:

ansible all -b -m file -a "state=absent path=/opt/enigma/tmp/deployment-passwords.txt"

## Conclusion

## Congratulations, you have completed this hands-on lab!

## Additional Resources

The development team has released the latest build of the enigma software. You have been approached to install the latest version on the QA systems. The development team has provided a task list of how to install the build. Rather than run through the steps on all the servers in your environment, you can use Ansible to run each task on all the servers at one time!

The Ansible control node has been configured for you and each QA server has already been configured for use with Ansible. The default inventory has been configured to include a qa-servers host group which includes the host that requires the new software. Execute the task list provided below using the appropriate Ansible ad-hoc commands.

*SPECIAL NOTE:* Although you will initially log in as the cloud\_user, please sudo su - to the ansible user in order to complete the solution. All the SSH keys have been created and exchanged among the hosts in order to allow the tasks to be run appropriately.

**Note:** This URL only works correctly when accessed on the lab servers. The domain name DNS entry is overridden in /etc/hosts on each lab server. If you attempt to access it from another system, you will reach a parked domain.

Installation task list:

* Download <http://software.xyzcorp.com/enigma.tgz> to /tmp on each host in qa-servers and verify the sha256 checksum via <http://software.xyzcorp.com/enigma-checksum.txt>. (**Note:** There is a bug in ansible get\_url that currently only allows using the literal checksum value instead of a file location for the checksum argument! Thus, you must copy the checksum value from the provided checksum file and then paste that value into the ansible command rather than simply providing the checksum file URL in the command.)
* Extract /tmp/enigma.tgz to /opt/ on all hosts in qa-servers.
* Update the line of text "DEPLOY\_CODE" in /opt/enigma/details.txt to the "CODE\_RED" on each server in qa-servers.
* Set the group ownership of the directory /opt/enigma/secret/ and each file contained within the directory so that the group owner is protected for each host in qa-servers.
* Delete the file /opt/enigma/tmp/deployment-passwords.txt from all servers.

## Learning Objectives

0 of 5 completed

Become `ansible` user and then download <http://software.xyzcorp.com/enigma.tgz> to `/tmp` on each host in qa-servers and verify the sha256 checksum via <http://software.xyzcorp.com/enigma-checksum.txt>.

**Note:** This URL only works correctly when accessed on the lab servers. The domain name DNS entry is overridden in /etc/hosts on each lab server. If you attempt to access it from another system, you will reach a parked domain.

On EACH of the hosts in 'qa-servers' inventory, become the ansible user:

sudo su - ansible

As the ansible user (above), run the following commands on each host in the qa-servers:

* CHECKSUM=$(curl http://software.xyzcorp.com/enigma-checksum.txt | cut -f1 -d' ')
* ansible qa-servers -m get\_url -a "url=http://software.xyzcorp.com/enigma.tgz dest=/tmp/enigma.tgz checksum=sha256:$CHECKSUM"

Extract `/tmp/enigma.tgz` to `/opt/` on all hosts in `qa-servers`.

Run ansible qa-servers -b -m unarchive -a "src=/tmp/enigma.tgz dest=/opt/ remote\_src=yes".

Update the line of text "DEPLOY\_CODE" in `/opt/enigma/details.txt` to the "CODE\_RED" on each server in `qa-servers`.

Run ansible qa-servers -b -m lineinfile -a "regexp=DEPLOY\_CODE line=CODE\_RED path=/opt/enigma/details.txt".

Set the group ownership of the directory `/opt/enigma/secret/` and each file contained within the directory so that the group owner is `protected` for each host in `qa-servers`.

Run ansible qa-servers -b -m file -a "recurse=yes state=directory path=/opt/enigma/secret group=protected".

Delete the file `/opt/enigma/tmp/deployment-passwords.txt` from all servers.

Run ansible all -b -m file -a "state=absent path=/opt/enigma/tmp/deployment-passwords.txt".

CONTROL NODE :

[root@control cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/bin/echo 'S+|QdvN1' | /bin/passwd root —stdin

/bin/echo 'S+|QdvN1' | /bin/passwd cloud\_user —stdin

/bin/echo "StrictHostKeyChecking no" >> /etc/ssh/ssh\_config

sed -i s/requiretty/\!requiretty/ /etc/sudoers

hostnamectl set-hostname control

/bin/echo 10.0.1.163 software.xyzcorp.com node1>> /etc/hosts

/bin/echo 10.0.1.188 node2>> /etc/hosts

yum install -y centos-release-ansible-27

yum install -y ansible

yum install -y sshpass

/sbin/useradd ansible/bin/echo 'S+|QdvN1' | /bin/passwd ansible —stdin

/bin/echo "ansible ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

sudo -u ansible /bin/mkdir -p /home/ansible/.ssh

sudo -u ansible /bin/ssh-keygen -q -N "" -f /home/ansible/.ssh/id\_rsa

sudo -u ansible sshpass -p 'S+|QdvN1' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@localhost

sudo -u ansible sshpass -p 'S+|QdvN1' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@node1

sudo -u ansible sshpass -p 'S+|QdvN1' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@node2

/sbin/groupadd protected

echo "[qa-servers]" >> /etc/ansible/hostsecho "localhost" >> /etc/ansible/hosts

echo "node1" >> /etc/ansible/hosts

echo "node2" >> /etc/ansible/hosts

[root@control cloud\_user]# ansible —versionansible 2.7.18

config file = /etc/ansible/ansible.cfg

configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']

ansible python module location = /usr/lib/python2.7/site-packages/ansible

executable location = /bin/ansible

python version = 2.7.5 (default, Nov 16 2020, 22:23:17) [GCC 4.8.5 20150623 (Red Hat 4.8.5-44)]

[root@control cloud\_user]# cat /etc/os-release

NAME="CentOS Linux"

VERSION="7 (Core)"ID="centos"

ID\_LIKE="rhel fedora"

VERSION\_ID="7"

PRETTY\_NAME="CentOS Linux 7 (Core)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:centos:centos:7"

HOME\_URL="<https://www.centos.org/>“

BUG\_REPORT\_URL="<https://bugs.centos.org/>“

CENTOS\_MANTISBT\_PROJECT="CentOS-7"

CENTOS\_MANTISBT\_PROJECT\_VERSION="7"

REDHAT\_SUPPORT\_PRODUCT="centos"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="7"

[root@control cloud\_user]#

NODE 1 NODE 2 :

[root@node1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/bin/echo 'S+|QdvN1' | /bin/passwd cloud\_user —stdin

/bin/echo "StrictHostKeyChecking no" >> /etc/ssh/ssh\_config

/usr/bin/hostnamectl set-hostname node1

/bin/echo 127.0.0.1 software.xyzcorp.com node1>> /etc/hosts

/sbin/useradd ansible

/bin/echo 'S+|QdvN1' | /bin/passwd ansible —stdin

/bin/echo "ansible ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

/sbin/groupadd protected

yum install -y httpd

systemctl start httpd

systemctl enable httpd/sbin/setenforce 0

/bin/firewall-cmd —permanent —add-service=http

/bin/firewall-cmd —reload

/sbin/setenforce 1

/bin/mkdir /tmp/enigma

/bin/mkdir /tmp/enigma/tmp

/bin/mkdir /tmp/enigma/secret

/bin/mkdir /tmp/enigma/lib

/bin/echo This is a secrete file>/tmp/enigma/secret/message.txt

/bin/echo This is a secrete file>/tmp/enigma/secret/run\_creds.txt

/bin/echo This is a secrete file>/tmp/enigma/secret/launch\_codes.bin

/bin/echo 1337hAckermAn>/tmp/enigma/tmp/deployment-passwords.txt/bin/echo This is a binary lib file>/tmp/enigma/lib/x509.so

/bin/echo This is a binary lib file>/tmp/enigma/lib/enigma.so

/bin/echo This is a binary lib file>/tmp/enigma/lib/stl.so

/bin/echo This is a binary lib file>/tmp/enigma/lib/nsa-glibc.so/bin/echo "#!/bin/sh">/tmp/enigma/enigma

/bin/echo "#This is the main binary">>/tmp/enigma/enigma

/bin/echo echo "You see nothing!">>/tmp/enigma/enigma

/bin/echo VERSION: 1.23.6>/tmp/enigma/details.txt

/bin/echo CODE\_NAME: FOILHAT>>/tmp/enigma/details.txt/bin/echo DEPLOY\_CODE: NULL>>/tmp/enigma/details.txt

/bin/echo ARCH: 64>>/tmp/enigma/details.txt

/bin/echo OS: LINUX>>/tmp/enigma/details.txt

/bin/chmod +x /tmp/enigma/enigma

cd /tmp; /bin/tar -czf /var/www/html/enigma.tgz enigma; sleep 5

cd /var/www/html; sha256sum enigma.tgz > enigma-checksum.txt

rm -rf /tmp/enigma[root@node1 cloud\_user]# ansible —versionbash: ansible: command not found

[root@node1 cloud\_user]# cat /etc/os-release

NAME="CentOS Linux"

VERSION="7 (Core)"

ID="centos"

ID\_LIKE="rhel fedora"

VERSION\_ID="7"

PRETTY\_NAME="CentOS Linux 7 (Core)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:centos:centos:7"

HOME\_URL="<https://www.centos.org/>“

BUG\_REPORT\_URL="<https://bugs.centos.org/>“

CENTOS\_MANTISBT\_PROJECT="CentOS-7"

CENTOS\_MANTISBT\_PROJECT\_VERSION="7"

REDHAT\_SUPPORT\_PRODUCT="centos"

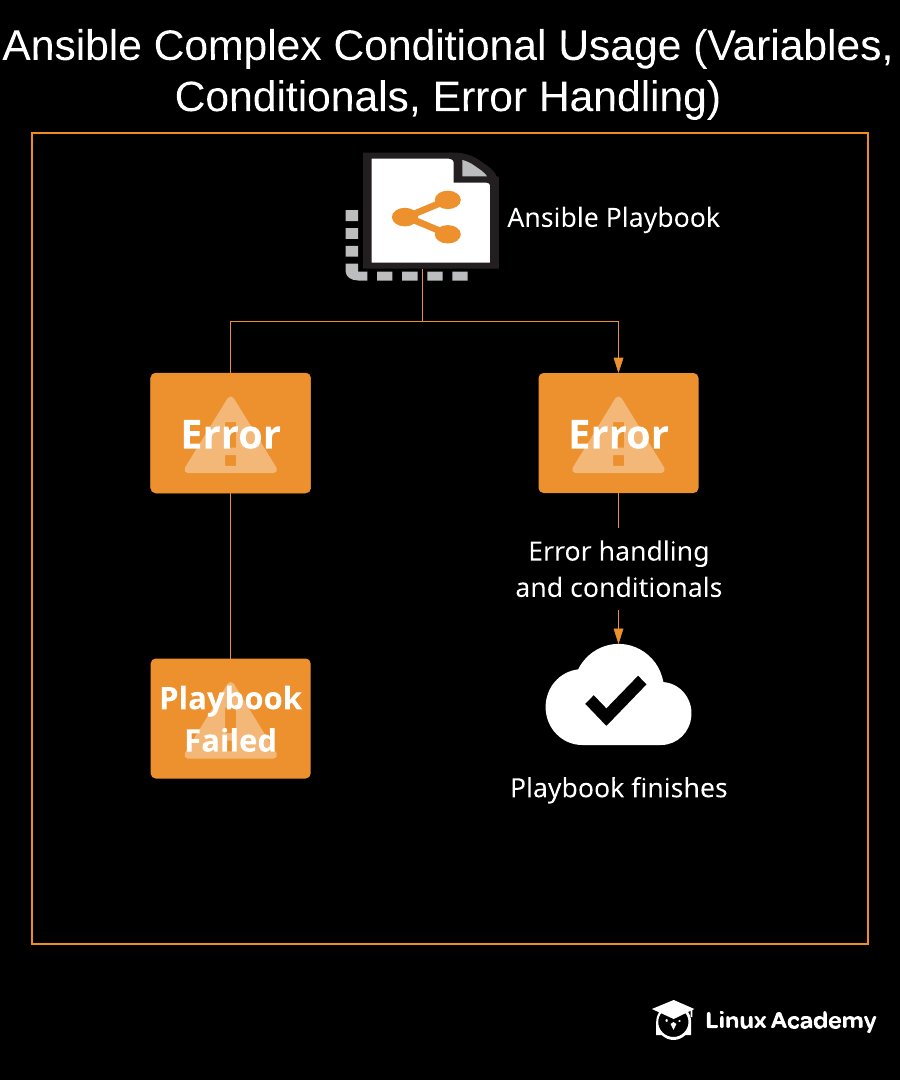
REDHAT\_SUPPORT\_PRODUCT\_VERSION="7"

.

[root@node1 cloud\_user]#

9

## Ansible Complex Conditional Usage (Variables, Conditionals, Error Handling)



# Ansible Complex Conditional Usage (Variables, Conditionals, Error Handling)

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## As with any technology, there's going to be errors or things that don't go exactly as planned. Learning how to handle errors as they happen, and building your playbooks to handle errors in a graceful fashion, is an important skill to have. This lab will help reinforce some of the ways to do this.

## The Scenario

## Our backup team wants us to install their backup package everywhere in the environment, and make sure that there's an archive file for them to back up, even if it's empty.

We know that the backup team's repository server has a very low uptime. We need to write our playbook so that installing the linuxacademy\_backup\_package will not cause the entire playbook to fail. In addition, we have to check and make sure that /root/archive.tar exists, and then create it if it doesn't.

## Logging In

## Use the credentials provided on the hands-on lab page to get into Server1 to begin with. Since we need root privileges, let's just run sudo -i right off and become root.

## Install the linuxacademy\_backup\_package, Accounting for the Repository or Package Not Being Available

## Create the playbook (vim contitional.yml) and create a task to install the package:

---  
#Conditional Playbook  
  
- name: Conditional Playbook  
  hosts: all  
  become: yes  
  
  tasks:  
   - name: Install backup package  
     yum:  
      name: linuxacademy\_backup\_package  
      state: present  
     ignore\_errors: true

## Check for the Existence of /root/archive.tar and Create It If It Doesn't

## This part of the playbook should look something like this:

   - name: Check for archive file  
     command: ls /root/archive.tar  
     register: check\_result  
     ignore\_errors: true  
   - name: Touch archive file  
     file:  
      path: /root/archive.tar  
      state: touch  
     when: check\_result is failed

## Run the Playbook

## Now let's see what happens when we run this:

ansible-playbook conditional.yml

## Conclusion

## We'll see errors, but the playbook keeps on going, ignoring them. That's what we wanted. Congratulations!

## Additional Resources

Notice: Ansible is installed as the root user, so please work on all tasks after elevating to the root user.

Our backup team wants us to install their backup package everywhere in the environment, and make sure that there's an archive file for them to back up, even if it's empty.

We know that the backup team's repository server has a very low uptime. We need to write our playbook so that installing the linuxacademy\_backup\_package will not cause the entire playbook to fail. In addition, we have to check and make sure that /root/archive.tar exists, and then create it if it doesn't.

## Learning Objectives

0 of 2 completed

Install the linuxacademy\_backup\_package, Accounting for the Repository or Package Not Being Available

The task should look similar to the following:

   - name: Install backup package  
     yum:  
      name: linuxacademy\_backup\_package  
      state: present  
     ignore\_errors: true

Check for the Existence of /root/archive.tar and Create It If It Doesn't

This part of the playbook should look something like this:

  - name: Check for archive file  
    command: ls /root/archive.tar  
    register: check\_result  
    ignore\_errors: true  
  - name: Touch archive file  
    file:  
     path: /root/archive.tar  
     state: touch  
    when: check\_result is failed.

Ansible host :

[root@Server1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname Server1

/usr/bin/yum install wget git -y

/bin/echo 'source /root/ansible/hacking/env-setup -q' >> /root/.bashrc

/bin/ssh-keygen -N '' -t rsa -f ~/.ssh/id\_rsa

/bin/cat /root/.ssh/id\_rsa.pub >> /home/ansible/.ssh/authorized\_keys

/bin/echo 'python\_interpreter=auto' >> /etc/ansible/ansible.cfg

/bin/echo 'host\_key\_checking = False' >> /etc/ansible/ansible.cfg

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/privkey> -O /tmp/pkey

/bin/cat /tmp/pkey > /root/.ssh/id\_rsa

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/akey

/bin/cat /tmp/akey > /root/.ssh/id\_rsa.pub

/bin/cat /tmp/akey > /home/ansible/.ssh/id\_rsa.pub

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:)]%5T4a-' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2255-cf002977c6404fe5835a0820c8a47f91 —resource PublicInstance —region us-east-1

[root@Server1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.7 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.7"

PLATFORM\_ID="platform:el8"PRETTY\_NAME="Red Hat Enterprise Linux 8.7 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/red_hat_enterprise_linux/8/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.7REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.7"

[root@Server1 cloud\_user]# ansible —version

ansible 2.10.0.dev0

config file = /etc/ansible/ansible.cfg

configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']

ansible python module location = /root/ansible/lib/ansible

executable location = /root/ansible/bin/ansible

python version = 3.6.8 (default, Jan 11 2023, 08:43:50) [GCC 8.5.0 20210514 (Red Hat 8.5.0-16)]

[root@Server1 cloud\_user]#

WebServer1 instance.

[root@WebServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash/usr/bin/hostnamectl set-hostname WebServer1/usr/bin/yum install wget -y

/sbin/useradd ansible

/usr/bin/echo 'ansible:)]%5T4a-' | /usr/sbin/chpasswd

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/key/bin/mkdir /home/ansible/.ssh

/usr/bin/chmod 0700 /home/ansible/.ssh/

/bin/cat /tmp/key > /home/ansible/.ssh/authorized\_keys

/usr/bin/chmod 0600 /home/ansible/.ssh/authorized\_keys/usr/bin/chown ansible.ansible /home/ansible -R

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:)]%5T4a-' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz/usr/local/bin/cfn-signal> -e 0 —stack cfst-2255-cf002977c6404fe5835a0820c8a47f91 —resource PublicInstance2 —region us-east-1

[root@WebServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.9 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@WebServer1 cloud\_user]# ansible —versi.

bash: ansible: command not found...

Packages providing this file are:

'ansible-core'

'ansible'

[root@WebServer1 cloud\_user]#

DBSERVER :

[root@DBServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash/usr/bin/hostnamectl set-hostname DBServer1

/usr/bin/yum install wget -y/sbin/useradd ansible

/usr/bin/echo 'ansible:)]%5T4a-' | /usr/sbin/chpasswd/sbin/alternatives --set python /usr/bin/python3/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts/bin/cat /tmp/hosts > /etc/hosts/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/key/bin/mkdir /home/ansible/.ssh/usr/bin/chmod 0700 /home/ansible/.ssh//bin/cat /tmp/key > /home/ansible/.ssh/authorized\_keys

/usr/bin/chmod 0600 /home/ansible/.ssh/authorized\_keys

/usr/bin/chown ansible.ansible /home/ansible -R/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers/usr/bin/echo 'cloud\_user:)]%5T4a-' | /usr/sbin/chpasswd/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz/usr/local/bin/cfn-signal> -e 0 —stack cfst-2255-cf002977c6404fe5835a0820c8a47f91 —resource PublicInstance3 —region us-east-1

[root@DBServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"VERSION="8.9 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"HOME\_URL="https://www.redhat.com/"

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@DBServer1 cloud\_user]# ansible —versionbash: ansible: command not found...

ADMIN SERVER:

[root@AdminServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname AdminServer1/usr/bin/yum install wget -y

/sbin/useradd ansible

/usr/bin/echo 'ansible:)]%5T4a-' | /usr/sbin/chpasswd

/sbin/alternatives --set python /usr/bin/python3/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/key/bin/mkdir /home/ansible/.ssh

/usr/bin/chmod 0700 /home/ansible/.ssh/

/bin/cat /tmp/key > /home/ansible/.ssh/authorized\_keys

/usr/bin/chmod 0600 /home/ansible/.ssh/authorized\_keys

/usr/bin/chown ansible.ansible /home/ansible -R

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:)]%5T4a-' | /usr/sbin/chpasswd/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2255-cf002977c6404fe5835a0820c8a47f91 —resource PublicInstance4 —region us-east-1

[root@AdminServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.9 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"HOME\_URL="https://www.redhat.com/"

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

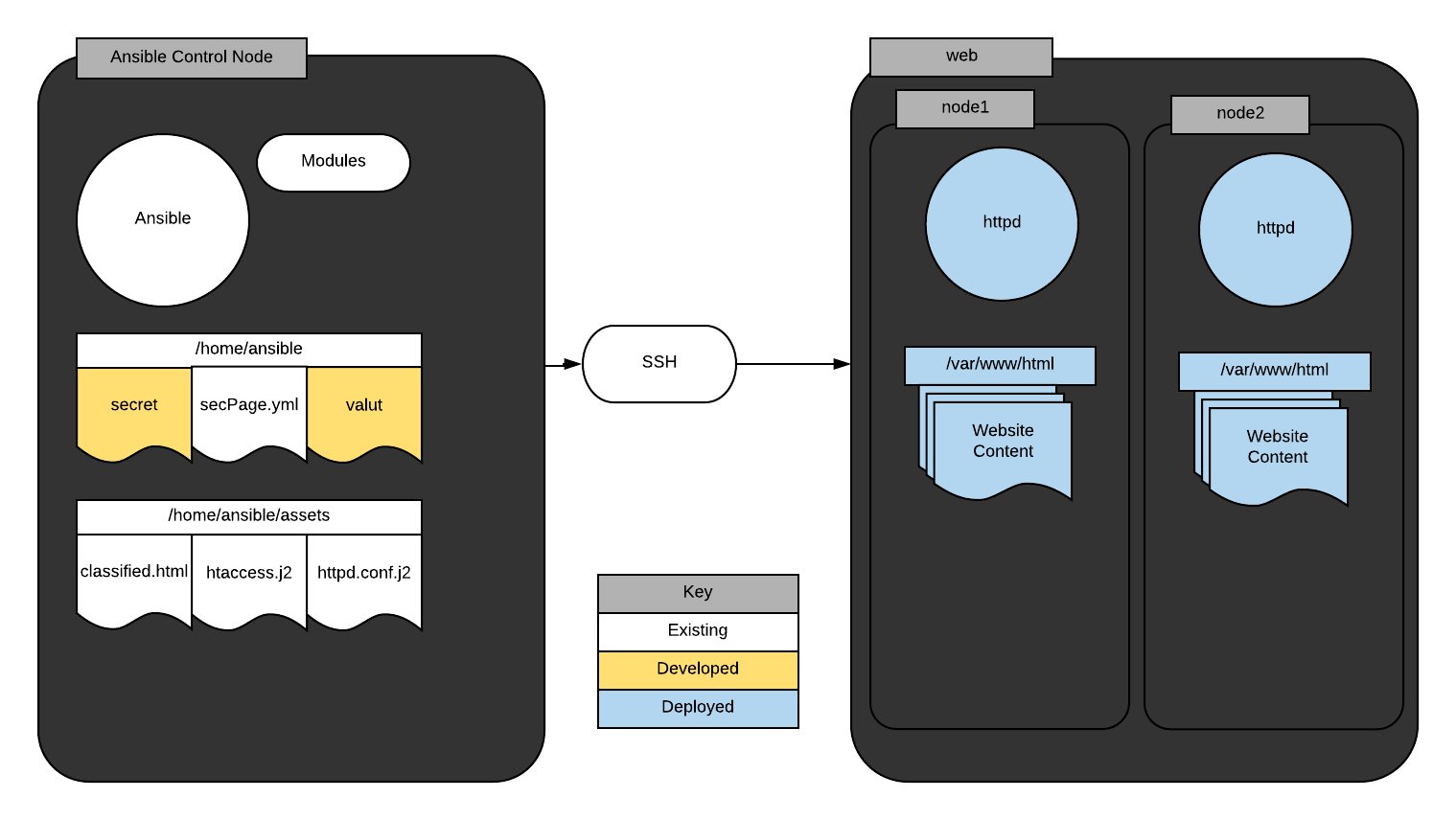
REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@AdminServer1 cloud\_user]# ansible —versionbash: ansible: command not found...

10 :

## Working with Confidential Data in Ansible.



# Working with Confidential Data in Ansible

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## The Red Hat Certified Ansible Specialist Exam (EX407) requires an understanding of working with confidential data within Ansible. This hands-on lab goes over how you can use the ansible-vault command to encrypt sensitive files within a vault file and also how to work with those vault files in an Ansible playbook. The exercise assumes basic proficiency with several common ansible modules and general ansible playbook use. Upon completing the lab, you will have developed an improved understanding of ansible-vault and vault files.

## Solution

## Log in to the Ansible Control Node via SSH:

ssh cloud\_user@<PUBLIC IP>

**Note:** Please give the lab an extra 5 minutes before connecting via SSH.

Switch to the ansible account (same password as the control node):

su ansible

### Encrypt /home/ansible/secret

Encrypt the file:

ansible-vault encrypt /home/ansible/secret

Give it an easy-to-remember new password, since we'll need it again later.

### Create a Vault Password File

Configure a vault password file named /home/ansible/vault to be used to run the Ansible playbook (replacing <YOUR VAULT PASSWORD> with the one you just created):

echo '<YOUR VAULT PASSWORD>' > /home/ansible/vault

### Run the Playbook

Run the playbook /home/ansible/secPage.yml using your vault password file to validate your work:

ansible-playbook --vault-password-file /home/ansible/vault /home/ansible/secPage.yml

### Verify the Secure Page Deployed Correctly

In the terminal, enter:

curl -u bond http://node1/secure/classified.html

When prompted for the password, enter james.

The command should return the contents of *classified.html* regarding the weather in a certain city.

## Conclusion

## Congratulations on successfully completing this hands-on lab!

## Additional Resources

**Note:** Please give the lab an extra 5 minutes before connecting via SSH.

In an effort to improve security, you have been tasked with securing an Ansible variable file. The variable file is to be used in an ansible job that creates a secure website. To do this, complete the following:

1. Encrypt the file */home/ansible/secret using ansible-vault*.
2. Then configure a vault password file named */home/ansible/vault* to be used to run the Ansible playbook */home/ansible/secPage.yml* successfully with the encrypted secrets file.
3. Verify your work by running the *secPage.yml* playbook using *ansible-playbook* and specifying your vault password file.
4. Test that the site deployed correctly by trying to access [*http://node1/secure/classified.html*](http://node1/secure/classified.html) using the user *bond* with the password *james*.

Summary tasks list:

* Encrypt *home/ansible/secret* using the ansible-vault command.
* Create */home/ansible/vault* as a vault password file that may be used to access the encrypted secret file without prompt.
* Run the playbook */home/ansible/secPage.yml* using your vault password file to validate your work.
* Verify that the secure page deployed correctly by attempting to access <http://node1/secure/classified.html> as the user *bond* with the password *james*.

Important notes:

* For your convenience, Ansible has been installed on the control node.
* */home/ansible/secPage.yml* provides several examples of how to use various key ansible modules. It is worth looking at, but it should not require modification.
* The user *ansible* is already on all servers with the appropriate shared keys for access to all necessary servers from the control node.
* The *ansible* user also has sudo access with no password. It has the same password as *cloud\_user*.
* All necessary Ansible inventories have been created for you.

## Learning Objectives

0 of 4 completed

Encrypt `/home/ansible/secret` using the `ansible-vault` command.

**Note:** Please give the lab an extra 5 minutes before connecting via SSH.

* Switch to the ansible user with su ansible
* Run ansible-vault encrypt /home/ansible/secret and provide a simple password of your choosing.
* Be sure to remember the password!

Create \*/home/ansible/vault\* as a vault password file that may be used to access the encrypted secret file without prompt.

* Run the command echo '<Your\_Vault\_Password>' > /home/ansible/vault.
* Substitute <<Your\_Vault\_Password>Your\_Vault\_Password> with the password you chose in the previous task.

Run the playbook \*/home/ansible/secPage.yml\* using your \*vault\* password file to validate your work.

* Run the command ansible-playbook --vault-password-file /home/ansible/vault /home/ansible/secPage.yml.
* If your encryption was configured correctly, you should get no errors.

Verify that the secure page deployed correctly by attempting to access <http://node1/secure/classified.html> as the user \*bond\* with the password \*james\*.

* Run curl -u bond http://node1/secure/classified.html and supply the password *james* when prompted.
* The command should return the contents of *classified.html* regarding the weather in a certain city.

CONTROL :

[root@control1 cloud\_user]# curl 169.254.169.254/latest/user-data#!/bin/bash

/bin/echo '+Vz(CjV9' | /bin/passwd cloud\_user —stdin/bin/echo "StrictHostKeyChecking no" >> /etc/ssh/ssh\_configsed -i s/requiretty/\!requiretty/ /etc/sudoershostnamectl set-hostname control1/bin/echo 10.0.1.241 node1>> /etc/hosts/bin/echo 10.0.1.44 node2>> /etc/hosts

yum-config-manager —enable rhui-REGION-rhel-server-extras

cd /tmp/usr/bin/yum install -y <https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm/usr/bin/yum> install -y ansible sshpass git python-passlib/sbin/useradd ansible/bin/echo '+Vz(CjV9' | /bin/passwd ansible —stdin/bin/echo "ansible ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

sudo -u ansible /bin/mkdir -p /home/ansible/.ssh

sudo -u ansible /bin/ssh-keygen -q -N "" -f /home/ansible/.ssh/id\_rsasudo -u ansible sshpass -p '+Vz(CjV9' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@localhost

sudo -u ansible sshpass -p '+Vz(CjV9' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@node1

sudo -u ansible sshpass -p '+Vz(CjV9' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@node2echo "localhost" >> /etc/ansible/hosts

echo "[webservers]" >> /etc/ansible/hostsecho "node1" >> /etc/ansible/hosts

echo "node2" >> /etc/ansible/hostsgit clone <https://github.com/linuxacademy/content-rh-ansible.git> /root/content-rh-ansible/bin/mv /root/content-rh-ansible/lab8/\* /home/ansible/

/bin/rm -f /home/ansible/ansible\_lab\*

chown -R ansible:ansible /home/ansible

/bin/rm -rf /root/content-rh-ansible

[root@control1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux Server"VERSION="7.9 (Maipo)"

ID="rhel"

ID\_LIKE="fedora"VARIANT="Server"

VARIANT\_ID="server"

VERSION\_ID="7.9"

PRETTY\_NAME="Red Hat Enterprise Linux Server 7.9 (Maipo)"

ANSI\_COLOR="0;31"CPE\_NAME="cpe:/o:redhat:enterprise\_linux:7.9:GA:server"

HOME\_URL="<https://www.redhat.com/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 7"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=7.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="7.9"

[root@control1 cloud\_user]# ansible —version

ansible 2.9.27

config file = /etc/ansible/ansible.cfg

configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']

ansible python module location = /usr/lib/python2.7/site-packages/ansible

executable location = /bin/ansible

python version = 2.7.5 (default, May 27 2022, 11:27:32) [GCC 4.8.5 20150623 (Red Hat 4.8.5-44)]

[root@control1 cloud\_user]#

NODE :

[root@node1 cloud\_user]# curl 169.254.169.254/latest/user-data#!/bin/bash

/bin/echo '+Vz(CjV9' | /bin/passwd cloud\_user —stdin

/bin/echo "StrictHostKeyChecking no" >> /etc/ssh/ssh\_config/usr/bin/hostnamectl set-hostname node1/usr/bin/yum-config-manager —enable rhui-REGION-rhel-server-extras

/sbin/useradd ansible

/bin/echo '+Vz(CjV9' | /bin/passwd ansible —stdin

/bin/echo "ansible ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

cd /tmp

/usr/bin/yum install -y <https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm[root@node1> cloud\_user]# cat /etc/os-releaseNAME="Red Hat Enterprise Linux Server"

VERSION="7.9 (Maipo)"ID="rhel"

ID\_LIKE="fedora"

VARIANT="Server"

VARIANT\_ID="server"

VERSION\_ID="7.9"

PRETTY\_NAME="Red Hat Enterprise Linux Server 7.9 (Maipo)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:7.9:GA:server"

HOME\_URL="<https://www.redhat.com/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 7"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=7.9REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

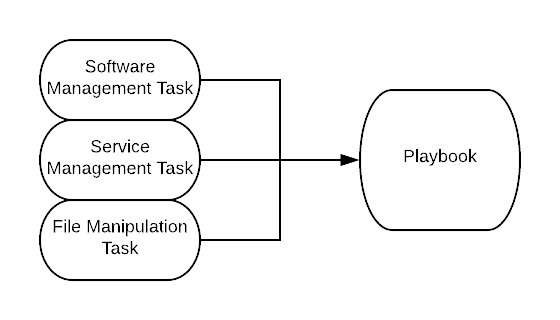
REDHAT\_SUPPORT\_PRODUCT\_VERSION="7.9"

[root@node1 cloud\_user]# ansible —versionbash: ansible: command not found

[root@node1 cloud\_user]#

11

## Understand Core Components of Ansible - Playbooks



# Understand Core Components of Ansible - Playbooks

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## Ansible is all about automation and making an administrator's life easier. So rather than chaining ad hoc commands together, you can set up a Playbook to run multiple commands at once. This hands-on lab will help enhance your understanding of building a playbook.

## The Scenario

## We're in the midst of a proof of concept project. Now, we've been tasked with writing a playbook that will set our machine up as a web server, with the normal software that our company's web servers typically have running on them. The playbook must install the following software:

httpd  
git  
tcpdump  
php

Then we need to set up a second playbook that creates the following users:

security  
devs  
admins

All of those users should be members of the web group.

## Logging In

## Use the credentials provided on the hands-on lab page to get into Server1 to begin with. Since we need root privileges, let's just run sudo -i right off and become root.

## Write and Execute the Playbook to Install Software

## Using whatever text editor is most comfortable, create software.yml in the /root directory (we'll use vim here):

vim software.yml

The playbook should look something like this when we're done:

---  
# Software playbook  
#  
  
- name: Install required software  
  hosts: all  
  
  tasks:  
   - yum:  
      name: "{{ packages }}"  
      state: present  
     vars:  
       packages:  
        - httpd  
        - git  
        - tcpdump  
        - php  
  
   - service:  
      name: httpd  
      state: started  
      enabled: yes

Now we can run the playbook with ansible-playbook software.yml --become. Alternatively, we could have stuck a become: yes line below the hosts line, and skipped having to put it in the command. If we do that though, we need to remember this is YAML, and very finicky about spacing. The become line needs to have the exact indentation that the hosts line does.

After a quick authenticity prompt, the playbook will run. We can see all of the software install and fire up. If we run the command again, we'll just get a bunch of **ok:** messages, because there's nothing to do this time around.

## Write and Execute the Playbook to Add and Configure the Required Users

## Now let's make a second playbook, the one that will create the users. Create users.yml in the /root directory ( again we'll use vim here):

vim users.yml

Your playbook should look something like this:

---  
# Create required users  
  
- name: Create required users  
  hosts: all  
  vars:  
   usernames:    
    - devs  
    - security  
    - admins  
  
  
  tasks:  
   - group:  
      name: web  
      state: present  
  
   - user:  
      name: "{{ item }}"  
      state: present  
      groups: web  
      append: yes  
     with\_items: "{{ usernames }}"

Just like with the last playbook, we'll run it with an ansible-playbook command:

ansible-playbook users.yml --become

Our output should show things that happened. We can check with id admins (which will print out a list of all the groups that admins is a member of). We can check the other users the same way.

## Conclusion

## We've successfully created a couple of playbooks. One installs software, and the other creates some users and adds them to groups. We are done. Congratulations!

## Additional Resources

Notice: Ansible is installed as the root user, so please work on all tasks after elevating to the root user.

We're in the midst of a proof of concept project. Now, we've been tasked with writing a playbook that will set our machine up as a web server, with the normal software that our company's web servers typically have running on them. The playbook must install the following software:

httpd  
git  
tcpdump  
php

Then we need to set up a second playbook that creates the following users:

security  
devs  
admins

All of those users should be members of the web group.

## Learning Objectives

0 of 2 completed

Write and Execute the Playbook to Install Software

Your playbook should look something like this:

---  
# Software playbook  
#  
  
- name: Install required software  
  hosts: all  
  
  tasks:  
   - yum:   
      name: "{{ packages }}"  
      state: present  
     vars:  
       packages:  
        - httpd   
        - git  
        - tcpdump  
        - php  
  
   - service:   
      name: httpd  
      state: started  
      enabled: yes

You can then run the command with ansible-playbook software.yml --become

Write and Execute the Playbook to Add and Configure the Required Users

Your playbook should look something like this:

---  
# Create required users  
  
- name: Create required users  
  hosts: all  
  vars:  
   usernames:    
    - devs  
    - security  
    - admins  
  
  
  tasks:  
   - group:  
      name: web  
      state: present  
  
   - user:  
      name: "{{ item }}"  
      state: present  
      groups: web  
      append: yes  
     with\_items: "{{ usernames }}"

Just like with the last playbook, we'll run it with an ansible-playbook command:

ansible-playbook users.yml --become

ONLY ONE VM PROVIDED :

[root@Server1 cloud\_user]# curl 169.254.169.254/latest/user-data#!/bin/bash/usr/bin/hostnamectl set-hostname Server1/bin/echo 'source /root/ansible/hacking/env-setup -q' >> /root/.bashrc

/bin/ssh-keygen -N '' -t rsa -f ~/.ssh/id\_rsa/bin/cat /root/.ssh/id\_rsa.pub >> /home/ansible/.ssh/authorized\_keys

/bin/echo 'python\_interpreter=auto' >> /etc/ansible/ansible.cfg

/sbin/alternatives —set python /usr/bin/python3

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:6eRxD6i)' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2222-c46990e7fffcb8d488b5c873341f1f18 —resource PublicInstance —region us-east-1

[root@Server1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"VERSION="8.7 (Ootpa)"

ID="rhel"ID\_LIKE="fedora"

VERSION\_ID="8.7"

PLATFORM\_ID="platform:el8"PRETTY\_NAME="Red Hat Enterprise Linux 8.7 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/red_hat_enterprise_linux/8/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.7

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.7"

[root@Server1 cloud\_user]# ansible —versionansible 2.10.0.dev0

config file = /etc/ansible/ansible.cfg configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']

ansible python module location = /root/ansible/lib/ansible

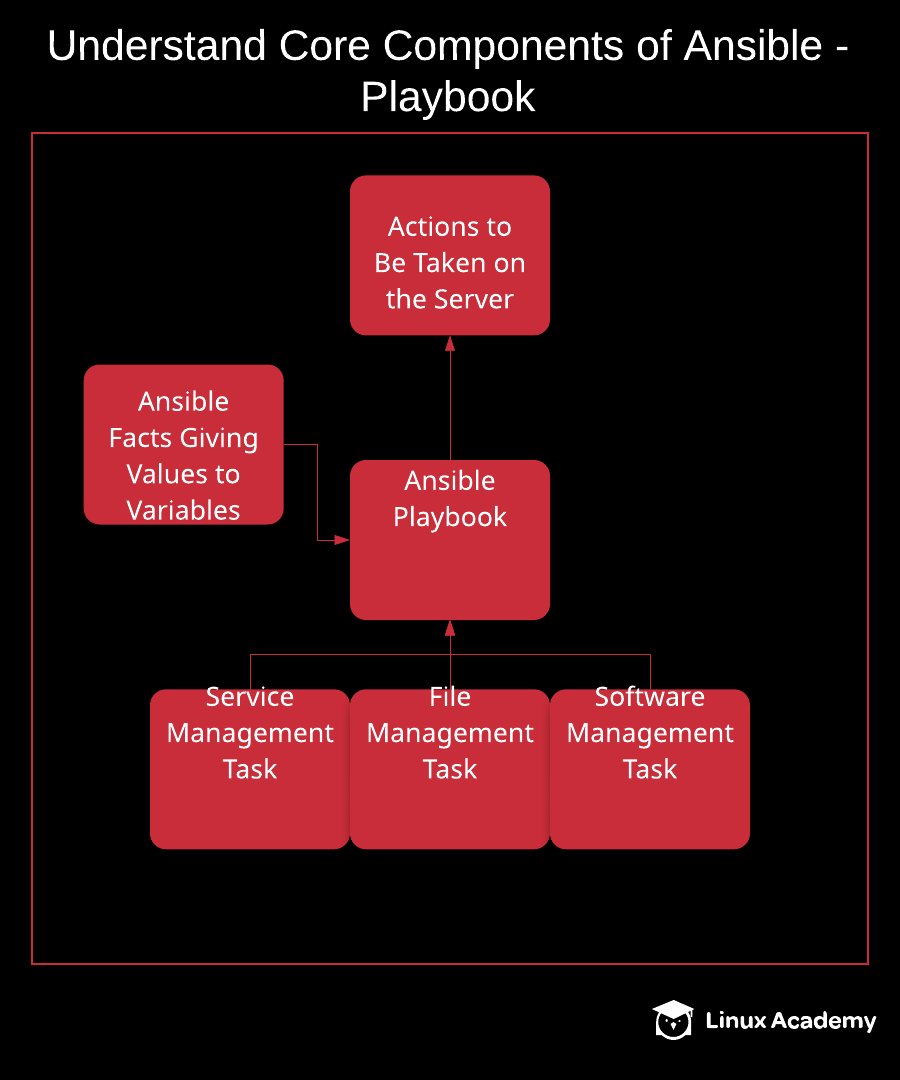
executable location = /root/ansible/bin/ansible

python version = 3.6.8 (default, Jan 11 2023, 08:43:50) [GCC 8.5.0 20210514 (Red Hat 8.5.0-16)]

[root@Server1 cloud\_user]#

12  :

## Understand Core Components of Ansible - Variables



# Understand Core Components of Ansible - Variables

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## Variables are another thing that makes Ansible powerful. By being able to write generic commands, with decision trees based on variables, we can automate almost anything. This lab will walk through a couple of things we can do with variables to help burn in the concept.

## The Scenario

## We're in the midst of a proof of concept project. Now we've got to make sure, on our existing setup, that tcpdump is only installed on servers with admin in their hostnames. And we want Git installed, but only on servers running RedHat. We can do it using Ansible variables.

## Logging In

## Use the credentials provided on the hands-on lab page to get into Server1 to begin with. Since we need root privileges, let's just run sudo -i right off and become root.

## Write a Playbook That Removes tcpdump When a Server's Name Does Not Contain admin

## Let's verify whether or not tcpdump is installed:

rpm -q tcpdump

Looks like it is. So now let's create the playbook (we'll use vim here, but any text editor is fine):

vim variables.yml

When we're done, the playbook should look something like this:

---  
# Variables playbook  
  
- name: This playbook will remove tcpdump (if installed) from servers without admin in their hostnames  
  hosts: all  
  become: yes  
  tasks:  
   - name: Remove tcpdump from all but admin servers  
     yum:  
      name: tcpdump  
      state: absent  
     when: "'admin' is not in inventory\_hostname"

Now we can run this playbook with:

ansible-playbook variables.yml

After an initial authenticity prompt, things should start moving. In the output, we'll see tcpdump get removed, because the hostname of the server (as it sits in the inventory) we're running it against doesn't contain admin.

## Modify That Playbook to Install Git on RedHat Servers

## Now let's get back into our playbook (vim variables.yml) and add the code that will install Git on our RedHat-based servers. Your additional lines should look something like this:

   - name: Make sure git is installed only on Red Hat servers  
     yum:  
      name: git  
      state: present  
     when: ansible\_facts['os\_family'] == 'RedHat'

Before we run the playbook though, let's see if Git is installed:

rpm -q git

It *is* installed. Let's get rid of it, then run the playbook and see what happens:

yum remove git -y  
ansible-playbook variables.yml

In the output, we'll see that something changed. Let's run rpm -q git again, and we'll see that the playbook installed the software correctly.

## Conclusion

## We were tasked with removing tcpdump from any server whose name did not include admin, and installing Git on any server that was running RedHat. We have a playbook that does both. We're finished. Congratulations!

## Additional Resources

Notice: Ansible is installed as the root user, so please work on all tasks after elevating to the root user.

We're in the midst of a proof of concept project. Now we've got to make sure, on our existing setup, that tcpdump is only installed on servers with admin in their hostnames. And we want Git installed, but only on servers running RedHat. We can do it using Ansible variables.

## Learning Objectives

0 of 2 completed

Write a Playbook That Removes `tcpdump` When a Server's Name Does Not Contain admin

Your playbook should look something like this:

---  
# Variables playbook  
  
- name: This playbook will remove tcpdump (if installed) from servers without admin in their hostnames  
  hosts: all  
  become: yes  
  tasks:  
   - name: Remove tcpdump from all but admin servers  
     yum:  
      name: tcpdump  
      state: absent  
     when: "'admin' not in inventory\_hostname"

Modify That Playbook to Install Git on RedHat Servers

Your additional lines should look something like this:

   - name: Make sure git is installed only on Red Hat servers  
     yum:  
      name: git  
      state: present  
     when: ansible\_facts['os\_family'] == 'RedHat'

ONE VM ONLY :

[root@Server1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname Server1

/bin/echo 'source /root/ansible/hacking/env-setup -q' >> /root/.bashrc

/bin/ssh-keygen -N '' -t rsa -f ~/.ssh/id\_rsa

/bin/cat /root/.ssh/id\_rsa.pub >> /home/ansible/.ssh/authorized\_keys

/bin/echo 'python\_interpreter=auto' >> /etc/ansible/ansible.cfg

/sbin/alternatives —set python /usr/bin/python3

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/yum install tcpdump -y

/usr/bin/echo 'cloud\_user:6IeVFVQ^' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2221-26504edfadfedeca47f31961582ba96d —resource PublicInstance —region us-east-1

[root@Server1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.7 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.7"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.7 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/red_hat_enterprise_linux/8/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.7REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.7"

[root@Server1 cloud\_user]# ansible —version

ansible 2.10.0.dev0 config file = /etc/ansible/ansible.cfg

configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']

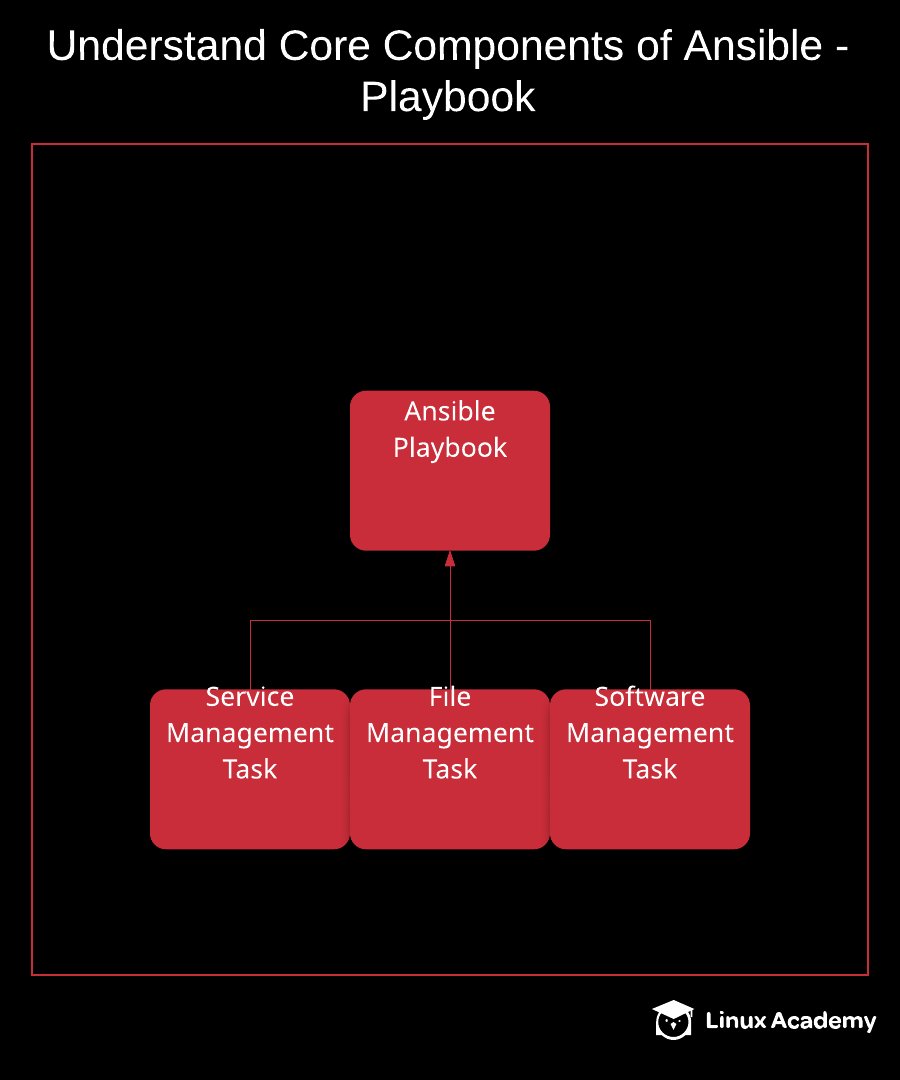
ansible python module location = /root/ansible/lib/ansible

executable location = /root/ansible/bin/ansible python version = 3.6.8 (default, Jan 11 2023, 08:43:50) [GCC 8.5.0 20210514 (Red Hat 8.5.0-16)]

[root@Server1 cloud\_user]#

13

## Understand Core Components of Ansible - Modules



# Understand Core Components of Ansible - Modules

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## Using built-in modules is one of the ways to extract a lot Ansible's power. In this lab, we go through managing file ownership, copying information, installing software, and managing services.

## The Scenario

## We're in the midst of a proof of concept project. Now we've been tasked with making sure that Ansible can create and manipulate files, as well as install software, on a target system. We need to install httpd and create an index.html file. The contents of that file will be server's facts. We also need to make sure all relevant services are started, and that the page is being served out. The index.html file should be owned by the apache user, have apache as the group owner, and have octal permissions of 0600.

## Logging In

## Use the credentials provided on the hands-on lab page to get into Server1 to begin with. Since we need root privileges, let's just run sudo -i right off and become root.

## Collect Facts

## If we're going to populate index.html with facts from the server, we need to collect them first:

ansible localhost -m setup --tree /tmp/facts

There will be an authenticity prompt, then the command will run. There should be a file in /tmp/facts now, called 127.0.0.1, that contains the facts.

## Install Apache

## Now, to serve out the facts we just collected, we've got to install Apache. But rather than using yum to do it, we're going to use Ansible:

ansible localhost -m yum -a "name=httpd state=present" --become

Remember, we need to use the --become flag so that Ansible knows to elevate and get root privileges.

## Copy the Server's Facts into index.html

## Using Ansible we can just copy the facts file into our new web server directory, and naming it index.html in the process:

ansible localhost -m copy -a "src=/tmp/facts/127.0.0.1 dest=/var/www/html/index.html mode=0600 owner=apache group=apache" --become

Again, we need to use --become because /var/www/html isn't writable unless we're the apache or root user, and we're trying to make the file so that it's owned by someone who isn't ansible.

Now let's check to make sure that we actually have an index.html file sitting in our web server's content directory:

ll /var/www/html/

It should be there.

## Make Sure Apache is Running

## We've installed Apache, but it probably isn't running by default. Check with this:

systemctl status httpd

We should see some output indicating that it is **inactive (dead)**. Luckily, we can start and manage services with Ansible. Let's use it now to start Apache:

ansible localhost -m service -a "name=httpd state=started enabled=yes" --become

If we run systemctl status httpd again, we can see that it is now running, and should start at boot. We should be all set.

## Prove It

## Not so fast! We want to be sure that we are in fact serving out our facts. Test it with this curl command:

curl localhost

All of our facts should fly by on the screen.

## Conclusion

## Just as we set out to do, we managed to use Ansible modules to get facts, install Apache, and serve out a web page containing those facts. Congratulations!

## Additional Resources

Notice: Ansible is installed as the root user, so please work on all tasks after elevating to the root user.

We're in the midst of a proof of concept project. Now we've been tasked with making sure that Ansible can create and manipulate files, as well as install software, on a target system. We need to install httpd and create an index.html file. The contents of that file will be server's facts.

We also need to make sure all relevant services are started, and that the page is being served out. The index.html file should be owned by the apache user, have apache as the group owner, and have octal permissions of 0600.

## Learning Objectives

0 of 4 completed

Collect Facts

If we're going to populate index.html with facts from the server, we need to collect them first.

ansible localhost -m setup --tree /tmp/facts

Install Apache

Rather than using yum to install httpd we're going to use Ansible.

ansible localhost -m yum -a "name=httpd state=present" --become

We need to use the --become flag so that Ansible knows to elevate for root priveleges.

Copy the Server's Facts into index.html

If we look in the directory we used in the first task, there's a file named 127.0.0.1 with all the server's facts. Using Ansible we can just copy that into index.html.

ansible localhost -m copy -a "src=/tmp/facts/127.0.0.1 dest=/var/www/html/index.html mode=0600 owner=apache group=apache" --become

We need to use --become because /var/www/html isn't writable unless you're the apache or root user and we're trying to make the file owned by someone who isn't ansible.

Make Sure Apache is Running

We've installed Apache, but it isn't running by default. Luckily we can start and manage services with Ansible.

ansible localhost -m service -a "name=httpd state=started enabled=yes" --become

Again, we need to use --become because only root can start a service.

[root@Server1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname Server1

/bin/echo 'source /root/ansible/hacking/env-setup -q' >> /root/.bashrc

/bin/ssh-keygen -N '' -t rsa -f ~/.ssh/id\_rsa

/bin/cat /root/.ssh/id\_rsa.pub >> /home/ansible/.ssh/authorized\_keys

/bin/echo 'python\_interpreter=auto' >> /etc/ansible/ansible.cfg

/sbin/alternatives —set python /usr/bin/python3

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:\*F88tRrA' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2220-92d4c2fb937f50897a72458404e1268e —resource PublicInstance —region us-east-1

[root@Server1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"VERSION="8.7 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.7"PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.7 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"HOME\_URL="https://www.redhat.com/"

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/red_hat_enterprise_linux/8/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.7

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.7"

[root@Server1 cloud\_user]# ansible —version

ansible 2.10.0.dev0

config file = /etc/ansible/ansible.cfg configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']

ansible python module location = /root/ansible/lib/ansible

executable location = /root/ansible/bin/ansible

python version = 3.6.8 (default, Jan 11 2023, 08:43:50) [GCC 8.5.0 20210514 (Red Hat 8.5.0-16)]

[root@Server1 cloud\_user]#

14

## Using Variables in Ansible Inventory Files

## ERROR UNABLE TO SSH



Using Variables in Ansible Inventory Files  
This course is not approved or sponsored by Red Hat.  
  
Introduction  
Cat Corp. is trialing a setup of Ansible in their environment. They've set up an Ansible host server and need to set up a couple of backup servers. You've been contracted to write a playbook that will install csh on the servers. Additionally, for this backup software, each server can run a different number of backup streams. This is configurable in a file located at /opt/backup/streams. You also need to populate this file with the value stored in the host-based variable managed in the Ansible inventory.  
  
Solution  
Log in to the server using the credentials provided:  
  
ssh cloud\_user@<PUBLIC\_IP\_ADDRESS>  
Note: Make sure to elevate to root once logged in. The reason for this is because of how the lab server’s version of Ansible is installed.  
  
Note: When copying and pasting code into Vim from the lab guide, first enter :set paste (and then i to enter insert mode) to avoid adding unnecessary spaces and hashes. To save and quit the file, press Escape followed by :wq. To exit the file without saving, press Escape followed by :q!.  
  
Use Ansible to Install csh  
Log in to the Instant Terminal and elevate yourself to root\_user:  
sudo -i  
List the contents of the directory:  
ls  
cat the inventory file:  
cat inventory  
Perform an Ansible ping:  
ansible -m ping -i inventory all  
Clear the screen:  
clear  
Install Vim for YAML:  
vim install\_software.yml  
Set the following values:  
---  
- hosts: all  
  become: yes  
  tasks:  
    - name: Install packages  
      yum:  
        state: present  
        name: csh  
Save and quit:  
:wq  
Install software:  
ansible-playbook -i inventory install\_software.yml  
Populate the File /opt/backups/streams with the Value Stored in the Host-Based Variable from the Inventory File  
Run the following query:  
mv install\_software.yml lab\_playbook.yml  
Edit file:  
vim lab\_playbook.yml  
Set the following values:  
---  
- hosts: all  
  become: yes  
  tasks:  
    - name: Install packages  
      yum:  
        state: present  
        name: csh  
    - name: Create required directory  
      file:  
        path: "/opt/backup"  
        state: directory  
    - name: Populate required file  
      copy:  
        dest: /opt/backup/streams  
        content: |  
          {{ streams }}  
Save and quit:  
:wq  
Clear your screen:  
clear  
Run the following query:  
ansible-playbook -i inventory lab\_playbook.yml  
SSH into backup1:  
ssh cloud\_user@backup1  
Run the following query:  
cat /opt/backup/streams  
Log out:  
logout  
cat the inventory file:  
cat inventory  
SSH into backup2:  
ssh cloud\_user@backup2  
Run the following query:  
cat /opt/backup/streams  
Log out:  
logout  
cat the playbook:  
cat lab\_playbook.yml  
Conclusion  
Congratulations — you've completed this hands-on lab!

## Additional Resources

There are 2 tasks to accomplish for this hands-on lab.

1. Write a playbook that installs csh on the provisioned servers.
2. Modify that playbook to populate the /opt/backup/streams file with the value of the host based-variable from the Ansible inventory file.

## Learning Objectives

0 of 2 completed

Use Ansible to Install csh

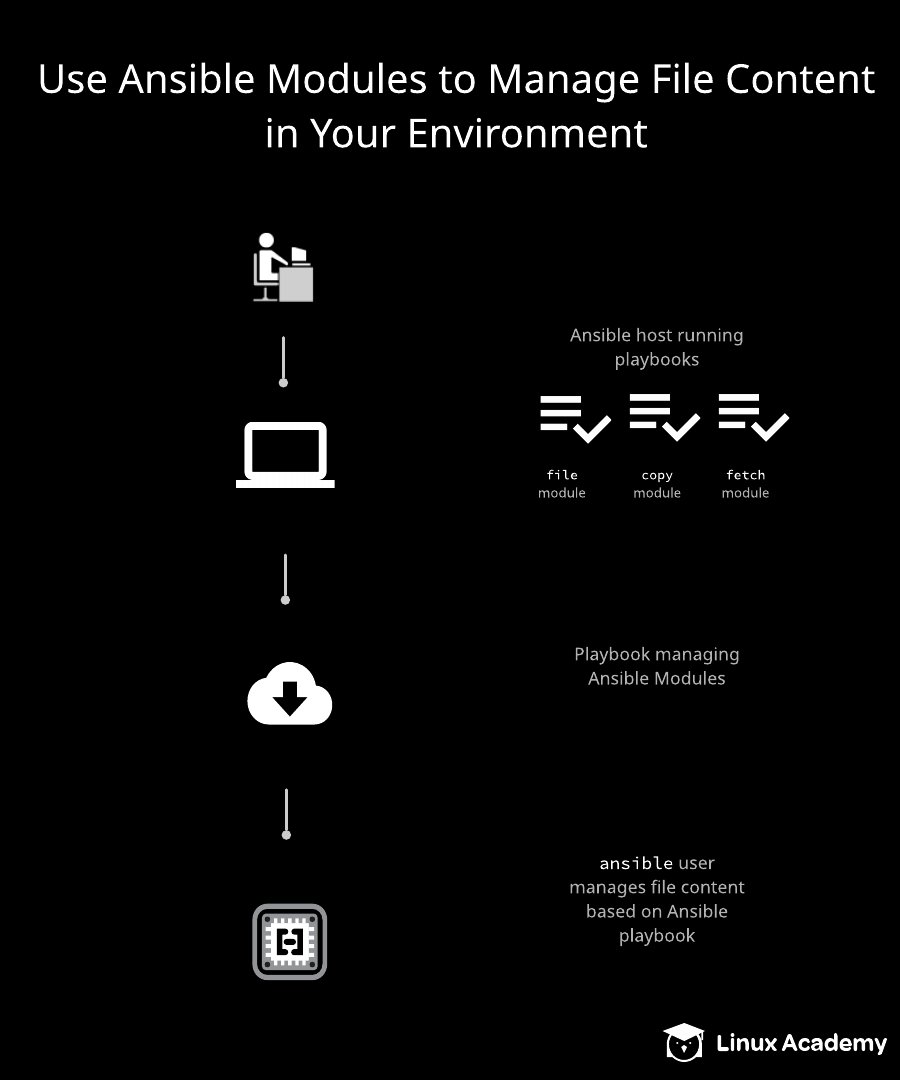
Write a playbook to install csh on the newly provisioned web server.

Populate the File `/opt/backups/streams` with the Value Stored in the Host-Based Variable from the Inventory File

Add to the playbook to create a text file in /opt/backup/streams/ and populate it with the value of the host-based variable in the Ansible inventory.

15

## Using Ansible Modules to Manage File Content in Your Environment



# Using Ansible Modules to Manage File Content in Your Environment

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## When your web devs come up with an update to an application and you need to deploy it to 100 servers (sometimes more) doing so manually is a difficult task. While there are many ways to automate this process Ansible can handle it in addition to everything else it does. This lab will walk through a method of doing so.

## Solution

## Log in to the Host instance server using the credentials provided:

ssh cloud\_user@<PUBLIC IP ADDRESS>

Then become root:

sudo su -

### Deploy index.html

Create a playbook:

vim filedeploy.yml

Enter the following:

---  
- name: copy webapp  
  hosts: webservers  
  become: yes  
  
  tasks:  
   - name: install webapp  
     copy:  
      src: /root/index.html  
      dest: /var/www/html/index.html  
      owner: apache  
      group: apache  
      mode: 0644

### Copy and Execute the db deploy Script

In the same YAML file, at the bottom, enter:

- name: copy db script  
  hosts: dbservers  
  become: yes  
  
  tasks:  
   - name: copy deploy script  
#    script: /root/deploy.sh  
     copy:  
      src: /root/deploy.sh  
      dest: /opt/deploy.sh  
      owner: root  
      group: root  
      mode: 0755  
  
   - name: Execute the script  
     command: sh /opt/deploy.sh

Save and quit the file.

Run the playbook:

ansible-playbook filedeploy.yml

Run the following:

curl webserver1

We should see "Hello Wrold!" This must be the bug our dev team fixed.

### Redeploy index.html

Reopen the playbook:

vim filedeploy.yml

Delete the db deploy in the playbook and change the src to /root/index2.html:

---  
  
- name: copy webapp  
 hosts: webservers  
 become: yes  
  
 tasks:  
  - name: install webapp  
    copy:  
     src: /root/index2.html  
     dest: /var/www/html/index.html  
     owner: apache  
     group: apache  
     mode: 0644

Save and quit the file.

Run the playbook:

ansible-playbook filedeploy.yml

Run the following:

curl webserver1

We should now see the correctly spelled "Hello World!"

## Conclusion

## Congratulations on successfully completing this hands-on lab!

## Additional Resources

Notice: Ansible is installed as the root user, so please work on all tasks after elevating to the root user.

You received a ticket to upgrade the application and database for servers in your environment. There is an index.html and deploy.sh script in the /root directory on your Ansible Host. Write a playbook that deploys the index.html file to the webserver (in /var/www/html/), then deploys deploy.sh script to the dbserver in /opt, and finally executes that deploy.sh script. Use curl to test your deployment.

Once that deployment has finished, your developers let you know they found and fixed a major bug, and this fix needs to be deployed immediately. Deploy the /root/index2.html file to address this issue.

## Learning Objectives

0 of 3 completed

Deploy index.html

Your playbook should look something like:

---  
- name: copy webapp  
  hosts: webservers  
  become: yes  
     
  tasks:  
   - name: install webapp  
     copy:  
      src: /root/index.html  
      dest: /var/www/html/index.html  
      owner: apache  
      group: apache  
      mode: 0644

Copy and Execute the `db deploy` (deploy.sh) Script

There are two different methods of doing this in a playbook.

The first is to use script, but it is commented out. Both can be seen in this code snippet.

- name: copy db script  
  hosts: dbservers  
  become: yes  
  
  tasks:   
   - name: copy deploy script  
#    script: /root/deploy.sh  
     copy:  
      src: /root/deploy.sh  
      dest: /opt/deploy.sh  
      owner: root  
      group: root  
      mode: 0755  
  
   - name: Execute the script  
     command: sh /opt/deploy.sh

Run the playbook with this:

ansible-playbook filedeploy.yml

Redeploy index.html

You should not include the db deploy in this playbook.

---  
  
- name: copy webapp  
 hosts: webservers  
 become: yes  
  
 tasks:  
  - name: install webapp  
    copy:  
     src: /root/index2.html  
     dest: /var/www/html/index.html  
     owner: apache  
     group: apache  
     mode: 0644

Run the playbook again:

ansible-playbook filedeploy.yml

HOST NODE :

[root@Server1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname Server1

/usr/bin/yum install wget git -y

/bin/echo 'source /root/ansible/hacking/env-setup -q' >> /root/.bashrc

/bin/ssh-keygen -N '' -t rsa -f ~/.ssh/id\_rsa/bin/cat /root/.ssh/id\_rsa.pub >> /home/ansible/.ssh/authorized\_keys

/bin/echo 'python\_interpreter=auto' >> /etc/ansible/ansible.cfg

/bin/echo 'host\_key\_checking = False' >> /etc/ansible/ansible.cfg

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/privkey> -O /tmp/pkey

/bin/cat /tmp/pkey > /root/.ssh/id\_rsa

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/akey

/bin/cat /tmp/akey > /root/.ssh/id\_rsa.pub

/bin/cat /tmp/akey > /home/ansible/.ssh/id\_rsa.pub

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/file-lab/deploy.sh> -O /root/deploy.sh

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/file-lab/index.html> -O /root/index.html

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/file-lab/index2.html> -O /root/index2.html/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:V6+twc&Z' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2272-1bbeef2ee4d031bc36a95d0776cac3ed —resource PublicInstance —region us-east-1

[root@Server1 cloud\_user]# cat /etc/os-releaseNAME="Red Hat Enterprise Linux"

VERSION="8.7 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.7"PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.7 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>"DOCUMENTATION\_URL="<https://access.redhat.com/documentation/red_hat_enterprise_linux/8/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.7

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.7"

[root@Server1 cloud\_user]# ansible —version

ansible 2.10.0.dev0

config file = /etc/ansible/ansible.cfg

configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']

ansible python module location = /root/ansible/lib/ansible

executable location = /root/ansible/bin/ansible

python version = 3.6.8 (default, Jan 11 2023, 08:43:50) [GCC 8.5.0 20210514 (Red Hat 8.5.0-16)]

[root@Server1 cloud\_user]#

NODE 1 WEBSERVER L:

[root@WebServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash/usr/bin/hostnamectl set-hostname WebServer1

/usr/bin/yum install wget httpd -y/sbin/useradd ansible

/usr/bin/echo 'ansible:V6+twc&Z' | /usr/sbin/chpasswd

/sbin/alternatives --set python /usr/bin/python3/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/key/bin/mkdir /home/ansible/.ssh/usr/bin/chmod 0700 /home/ansible/.ssh/

/bin/cat /tmp/key > /home/ansible/.ssh/authorized\_keys

/usr/bin/chmod 0600 /home/ansible/.ssh/authorized\_keys

/usr/bin/chown ansible.ansible /home/ansible -R

/usr/bin/systemctl start httpd

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers/usr/bin/echo 'cloud\_user:V6+twc&Z' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2272-1bbeef2ee4d031bc36a95d0776cac3ed —resource PublicInstance2 —region us-east-1

[root@WebServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.9 (Ootpa)"ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>"DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

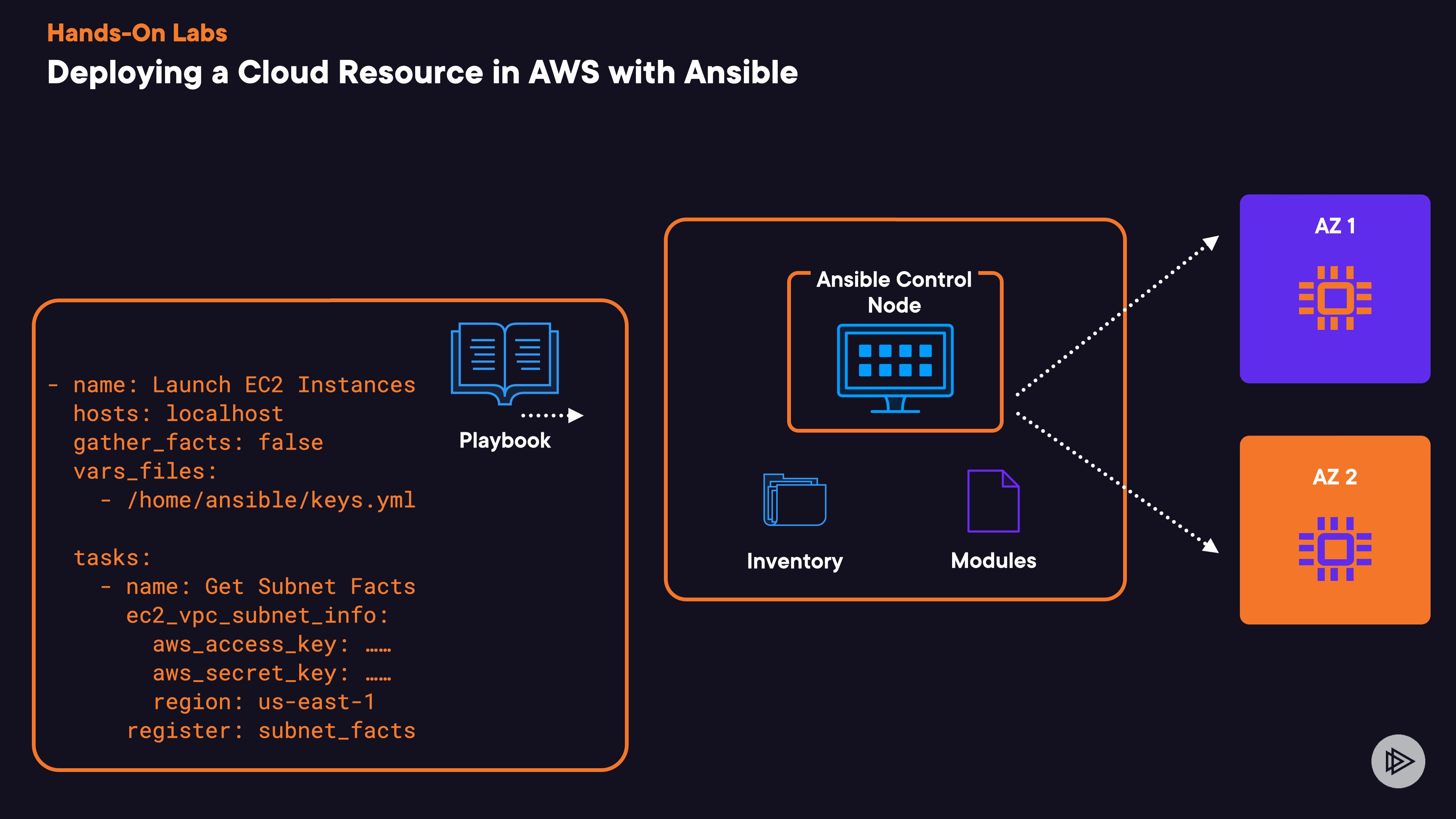
REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@WebServer1 cloud\_user]# ansible —version

bash: ansible: command not found...

16

## Deploying a Cloud Resource in AWS with Ansible



Deploying a Cloud Resource in AWS with Ansible  
Introduction  
In this lab, we will use Ansible to launch EC2 instances in different availability zones. The launch\_ec2\_instances.yml playbook defines three tasks: the first to get the subnet facts, the next to set the subnet IDs and availability zones, and a final task responsible for launching an EC2 instance into two specific subnets.  
  
Solution  
Log in to the server using the credentials provided:  
  
ssh cloud\_user@<PUBLIC\_IP\_ADDRESS>  
Create and Edit launch\_ec2\_instances.yml File  
Become the ansible user:  
  
su - ansible  
Create and edit the launch\_ec2\_instances playbook:  
  
sudo vim launch\_ec2\_instances.yml  
Paste in the following file information:  
  
- name: Launch EC2 Instances  
  hosts: localhost  
  gather\_facts: false  
  vars\_files:  
    - /home/ansible/keys.yml  
  
  tasks:  
    - name: Get Subnet Facts  
      ec2\_vpc\_subnet\_info:  
        aws\_access\_key: "{{ AWS\_ACCESS\_KEY\_ID }}"  
        aws\_secret\_key: "{{ AWS\_SECRET\_ACCESS\_KEY }}"  
        region: us-east-1  
      register: subnet\_facts  
  
    - name: Set Subnet IDs and Availability Zones  
      set\_fact:  
        subnet\_ids: "{{ subnet\_facts.subnets|map(attribute='id')|list }}"  
        availability\_zones: "{{ subnet\_facts.subnets|map(attribute='availability\_zone')|list }}"  
  
    - name: Launch EC2 Instances in Each Subnet  
      ec2:  
        aws\_access\_key: "{{ AWS\_ACCESS\_KEY\_ID }}"  
        aws\_secret\_key: "{{ AWS\_SECRET\_ACCESS\_KEY }}"  
        ec2\_region: us-east-1  
        instance\_type: t2.micro  
        image: ami-026ebd4cfe2c043b2  
        vpc\_subnet\_id: "{{ item }}"  
        wait: true  
      loop: "{{ subnet\_ids }}"  
In another window, log in to the AWS Management Console using the credentials provided on the lab instructions page. Make sure you're using the us-east-1 Region.  
  
Under Recently visited, select EC2.  
  
Click the Launch instance button.  
  
Select Launch instance.  
  
Under Application and OS Images (Amazon Machine Image), select Red Hat.  
  
Copy the AMI ID.  
  
Return to the terminal and paste the AMI ID into the image: line.  
  
Save and quit by pressing the Escape button on your keyboard and entering :wq.  
  
Execute the Playbook  
Launch the EC2 instances:  
  
ansible-playbook launch\_ec2\_instances.yml  
Confirm the Instances Have Been Deployed in AWS  
Return to the browser window or tab with the AWS console open.  
Select Instances (running). You should see the two instances are initializing.  
Conclusion  
Congratulations — you've completed this hands-on lab!

## Additional Resources

You work for an international e-commerce company that is expanding its operations to multiple Regions around the world. The company wants to set up its infrastructure in different AWS Availability Zones (AZs) to test services to its customers.

As the DevOps engineer responsible for infrastructure provisioning, you have decided to use Ansible to automate the deployment of EC2 instances in different AZs.

We will need to deploy EC2 instances meeting the following requirements:

* Type: **t2.micro**
* Region: **us-east-1**
* AMI: **Manually retrieve the AMI ID from the AWS Console**

From the Ansible Control node:

Create the playbook launch\_ec2\_instances.yml to perform the following tasks:

* + Get the subnet facts
  + Set the subnet IDs and availability zones
  + Deploy two EC2 instances meeting the described properties.

Execute the playbook launch\_ec2\_instances.yml

Validate that the instances deployed using the AWS console.

The Ansible control node has been configured and already has Ansible installed. The control node also has a system user named ansible configured with SSH access keys and necessary system privileges.

An IAM user ansible has been created on the provided AWS sandbox account. The access keys for the ansible IAM user are stored in /home/ansible/keys.sh and /home/ansible/keys.yml for whichever authentication method we prefer. The ansible IAM user has appropriate permissions to perform the required task.

The default Ansible inventory has been configured to include the Ansible control host as localhost.

## Learning Objectives

0 of 3 completed

Create and Edit launch\_ec2\_instances.yml File

After logging into the EC2 instance, run su - ansible to become the ansible user. The password is the same as it is for cloud\_user.

Create the playbook launch\_ec2\_instances.yml to perform the following tasks:

* Get the subnet facts
* Set the subnet IDs and Availability Zones
* Deploy two EC2 instances meeting the described properties.

We will need to deploy EC2 instances meeting the following requirements:

* Type: **t2.micro**
* Region: **us-east-1**
* AMI: **Manually retrieve the AMI ID from the AWS Console**

Execute the Playbook

Use ansible to execute the launch\_ec2\_instances.yml file created in the previous objective.

Confirm the Instances Have Been Deployed in AWS

Log in to the AWS console, and in the EC2 dashboard, confirm the new instance's existence and state.

[root@control cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/bin/echo 'G%6rA^T%' | /bin/passwd root —stdin

/bin/echo 'G%6rA^T%' | /bin/passwd cloud\_user —stdin

/bin/echo "StrictHostKeyChecking no" >> /etc/ssh/ssh\_config

sed -i s/requiretty/\!requiretty/ /etc/sudoers

hostnamectl set-hostname control

yum install -y epel-release

yum install -y ansible sshpass

yum install -y python-boto python-boto3

/sbin/useradd ansible

/bin/echo 'G%6rA^T%' | /bin/passwd ansible —stdin

/bin/echo "ansible ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoerssudo -u ansible /bin/mkdir -p /home/ansible/.ssh

sudo -u ansible /bin/ssh-keygen -q -N "" -f /home/ansible/.ssh/id\_rsa

sudo -u ansible sshpass -p 'G%6rA^T%' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@localhost

echo "localhost" >> /etc/ansible/hosts

ACCESS=AKIA6LZIGEI3BLWH46G2

SECRET=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

echo "export AWS\_ACCESS\_KEY\_ID='KEYACCESS'" > /home/ansible/accesstemplate

echo "export AWS\_SECRET\_ACCESS\_KEY='KEYSECRET'" > /home/ansible/secrettemplate

echo "#!/bin/sh" >> /home/ansible/keys.shsed -e "s|KEYACCESS|$ACCESS|" /home/ansible/accesstemplate >> /home/ansible/keys.sh

sed -e "s|KEYSECRET|$SECRET|" /home/ansible/secrettemplate >> /home/ansible/keys.shecho "export AWS\_REGION='us-east-1'" >> /home/ansible/keys.sh

chmod 700 /home/ansible/keys.sh

rm -f /home/ansible/accesstemplate /home/ansible/secrettemplate

echo "AWS\_ACCESS\_KEY\_ID: KEYACCESS" > /home/ansible/accesstemplate

echo "AWS\_SECRET\_ACCESS\_KEY: KEYSECRET" > /home/ansible/secrettemplatesed -e "s|KEYACCESS|$ACCESS|" /home/ansible/accesstemplate >> /home/ansible/keys.yml

sed -e "s|KEYSECRET|$SECRET|" /home/ansible/secrettemplate >> /home/ansible/keys.yml

echo "AWS\_REGION: us-east-1" >> /home/ansible/keys.yml

chmod 600 /home/ansible/keys.yml

rm -f /home/ansible/accesstemplate /home/ansible/secrettemplatechown ansible.ansible /home/ansible/\*

echo "finished" > /home/cloud\_user/.done

[root@control cloud\_user]# cat /etc/os-release

NAME="CentOS Linux"

VERSION="7 (Core)"

ID="centos"

ID\_LIKE="rhel fedora"

VERSION\_ID="7"

PRETTY\_NAME="CentOS Linux 7 (Core)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:centos:centos:7"

HOME\_URL="<https://www.centos.org/>“

BUG\_REPORT\_URL="<https://bugs.centos.org/>“

CENTOS\_MANTISBT\_PROJECT="CentOS-7"

CENTOS\_MANTISBT\_PROJECT\_VERSION="7"

REDHAT\_SUPPORT\_PRODUCT="centos"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="7"

[root@control cloud\_user]# ansible —version

ansible 2.9.27

config file = /etc/ansible/ansible.cfg

configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']

ansible python module location = /usr/lib/python2.7/site-packages/ansible

executable location = /bin/ansible

python version = 2.7.5 (default, Nov 16 2020, 22:23:17) [GCC 4.8.5 20150623 (Red Hat 4.8.5-44)]

[root@control cloud\_user]#

AWS CFN :

{  
 "AWSTemplateFormatVersion" : "2010-09-09",  
 "Description" : "Deploying EC2 Instances with Ansible",  
  
 "Mappings" : {  
  
   "SubnetConfig" : {  
     "VPC"     : { "CIDR" : "10.0.0.0/16" },  
     "Public1"  : { "CIDR" : "10.0.1.0/24" },  
     "Public2": { "CIDR": "10.0.2.0/24" }  
   }  
 },  
  
 "Resources" : {  
  
   "VPC" : {  
     "Type" : "AWS::EC2::VPC",  
     "Properties" : {  
  
       "EnableDnsSupport" : "true",  
       "EnableDnsHostnames" : "true",  
       "CidrBlock" : { "Fn::FindInMap" : [ "SubnetConfig", "VPC", "CIDR" ]},  
       "Tags" : [  
        { "Key" : "Name", "Value" : "LinuxAcademy" },  
         { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },  
         { "Key" : "Network", "Value" : "VPC" }  
       ]  
     }  
   },  
  
  
   "PublicSubnet1" : {  
     "Type" : "AWS::EC2::Subnet",  
         "Properties" : {  
       "VpcId" : { "Ref" : "VPC" },  
       "AvailabilityZone" : {  
         "Fn::Select" : [ "0", { "Fn::GetAZs" : "" } ]  
       },  
       "CidrBlock" : { "Fn::FindInMap" : [ "SubnetConfig", "Public1", "CIDR" ]},  
       "Tags" : [  
         { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },  
         { "Key" : "Network", "Value" : "Public1" }  
       ]  
     }  
   },  
  
    "PublicSubnet2" : {  
     "Type" : "AWS::EC2::Subnet",  
         "Properties" : {  
       "VpcId" : { "Ref" : "VPC" },  
       "AvailabilityZone" : {  
         "Fn::Select" : [ "1", { "Fn::GetAZs" : "" } ]  
       },  
       "CidrBlock" : { "Fn::FindInMap" : [ "SubnetConfig", "Public2", "CIDR" ]},  
       "Tags" : [  
         { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },  
         { "Key" : "Network", "Value" : "Public2" }  
       ]  
     }  
   },    
  
   "InternetGateway" : {  
     "Type" : "AWS::EC2::InternetGateway",  
     "Properties" : {  
       "Tags" : [  
         { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },  
         { "Key" : "Network", "Value" : "Public" }  
       ]  
     }  
   },  
  
   "GatewayToInternet" : {  
      "Type" : "AWS::EC2::VPCGatewayAttachment",  
      "Properties" : {  
        "VpcId" : { "Ref" : "VPC" },  
        "InternetGatewayId" : { "Ref" : "InternetGateway" }  
      }  
   },  
  
   "PublicRouteTable" : {  
     "Type" : "AWS::EC2::RouteTable",  
     "Properties" : {  
       "VpcId" : { "Ref" : "VPC" },  
       "Tags" : [  
         { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },  
         { "Key" : "Network", "Value" : "Public" }  
       ]  
     }  
   },  
  
   "PublicRoute" : {  
     "Type" : "AWS::EC2::Route",  
     "DependsOn" : "GatewayToInternet",  
     "Properties" : {  
       "RouteTableId" : { "Ref" : "PublicRouteTable" },  
       "DestinationCidrBlock" : "0.0.0.0/0",  
       "GatewayId" : { "Ref" : "InternetGateway" }  
     }  
   },  
  
   "PublicSubnetRouteTableAssociation1" : {  
     "Type" : "AWS::EC2::SubnetRouteTableAssociation",  
     "Properties" : {  
       "SubnetId" : { "Ref" : "PublicSubnet1" },  
       "RouteTableId" : { "Ref" : "PublicRouteTable" }  
     }  
   },  
  
   "PublicSubnetRouteTableAssociation2" : {  
     "Type" : "AWS::EC2::SubnetRouteTableAssociation",  
     "Properties" : {  
       "SubnetId" : { "Ref" : "PublicSubnet2" },  
       "RouteTableId" : { "Ref" : "PublicRouteTable" }  
     }  
   },  
  
   "PublicNetworkAcl" : {  
     "Type" : "AWS::EC2::NetworkAcl",  
     "Properties" : {  
       "VpcId" : { "Ref" : "VPC" },  
       "Tags" : [  
         { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },  
         { "Key" : "Network", "Value" : "Public" }  
       ]  
     }  
   },  
  
   "InboundHTTPPublicNetworkAclEntry" : {  
     "Type" : "AWS::EC2::NetworkAclEntry",  
     "Properties" : {  
       "NetworkAclId" : { "Ref" : "PublicNetworkAcl" },  
       "RuleNumber" : "100",  
       "Protocol" : "6",  
       "RuleAction" : "allow",  
       "Egress" : "false",  
       "CidrBlock" : "0.0.0.0/0",  
       "PortRange" : { "From" : "80", "To" : "80" }  
     }  
   },  
  
   "InboundHTTPSPublicNetworkAclEntry" : {  
     "Type" : "AWS::EC2::NetworkAclEntry",  
     "Properties" : {  
       "NetworkAclId" : { "Ref" : "PublicNetworkAcl" },  
       "RuleNumber" : "101",  
       "Protocol" : "6",  
       "RuleAction" : "allow",  
       "Egress" : "false",  
       "CidrBlock" : "0.0.0.0/0",  
       "PortRange" : { "From" : "443", "To" : "443" }  
     }  
   },  
  
   "InboundSSHPublicNetworkAclEntry" : {  
     "Type" : "AWS::EC2::NetworkAclEntry",  
     "Properties" : {  
       "NetworkAclId" : { "Ref" : "PublicNetworkAcl" },  
       "RuleNumber" : "102",  
       "Protocol" : "6",  
       "RuleAction" : "allow",  
       "Egress" : "false",  
       "CidrBlock" : "0.0.0.0/0",  
       "PortRange" : { "From" : "22", "To" : "22" }  
     }  
   },  
  
   "InboundEmphemeralPublicNetworkAclEntry" : {  
     "Type" : "AWS::EC2::NetworkAclEntry",  
     "Properties" : {  
       "NetworkAclId" : { "Ref" : "PublicNetworkAcl" },  
       "RuleNumber" : "103",  
       "Protocol" : "6",  
       "RuleAction" : "allow",  
       "Egress" : "false",  
       "CidrBlock" : "0.0.0.0/0",  
       "PortRange" : { "From" : "1024", "To" : "65535" }  
     }  
   },  
  
   "OutboundPublicNetworkAclEntry" : {  
     "Type" : "AWS::EC2::NetworkAclEntry",  
     "Properties" : {  
       "NetworkAclId" : { "Ref" : "PublicNetworkAcl" },  
       "RuleNumber" : "100",  
       "Protocol" : "6",  
       "RuleAction" : "allow",  
       "Egress" : "true",  
       "CidrBlock" : "0.0.0.0/0",  
       "PortRange" : { "From" : "0", "To" : "65535" }  
     }  
   },  
  
   "PublicSubnetNetworkAclAssociation1" : {  
     "Type" : "AWS::EC2::SubnetNetworkAclAssociation",  
     "Properties" : {  
       "SubnetId" : { "Ref" : "PublicSubnet1" },  
       "NetworkAclId" : { "Ref" : "PublicNetworkAcl" }  
     }  
   },  
  
   "PublicSubnetNetworkAclAssociation2" : {  
     "Type" : "AWS::EC2::SubnetNetworkAclAssociation",  
     "Properties" : {  
       "SubnetId" : { "Ref" : "PublicSubnet2" },  
       "NetworkAclId" : { "Ref" : "PublicNetworkAcl" }  
     }  
   },     
  
"EC2SecurityGroup" : {  
     "Type" : "AWS::EC2::SecurityGroup",  
     "Properties" : {  
       "GroupDescription" : "Enable access to the EC2 host",  
       "VpcId" : { "Ref" : "VPC" },  
       "SecurityGroupIngress" : [  
         { "IpProtocol" : "tcp", "FromPort" : "22",  "ToPort" : "22",  "CidrIp" : "0.0.0.0/0" },  
         { "IpProtocol" : "tcp", "FromPort" : "80",  "ToPort" : "80",  "CidrIp" : "0.0.0.0/0" },  
         { "IpProtocol" : "tcp", "FromPort" : "443",  "ToPort" : "443",  "CidrIp" : "0.0.0.0/0" },  
         { "IpProtocol" : "icmp", "FromPort" : "-1",  "ToPort" : "-1",  "CidrIp" : "0.0.0.0/0" }  
       ]  
     }  
   },  
  "SGBaseIngress": {  
     "Type": "AWS::EC2::SecurityGroupIngress",  
     "Properties": {  
       "GroupId": { "Ref": "EC2SecurityGroup" },  
       "IpProtocol": "tcp",  
       "FromPort": "80",  
       "ToPort": "80",  
       "SourceSecurityGroupId": { "Ref": "EC2SecurityGroup" }  
     }  
   },  
   "ansible": {  
     "Type": "AWS::IAM::User",  
     "Properties": {  
       "Path": "/",  
       "ManagedPolicyArns" : [ "arn:aws:iam::aws:policy/AmazonEC2FullAccess" ],  
       "UserName": "ansible"  
     }  
   },  
   "ansiblekey" : {  
   "Type" : "AWS::IAM::AccessKey",  
   "Properties" : {  
      "UserName" : { "Ref" : "ansible" }  
      }  
   },  
   "Control" : {  
       "Type" : "AWS::EC2::Instance",  
       "Properties" : {  
           "InstanceType" : "t3.micro",  
           "ImageId" : "ami-09eb574578ce81bb9",  
           "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [  
               "#!/bin/bash\n",  
               "/bin/echo 'G%6rA^T%' | /bin/passwd root --stdin\n",  
               "/bin/echo 'G%6rA^T%' | /bin/passwd cloud\_user --stdin\n",  
               "/bin/echo \"StrictHostKeyChecking no\" >> /etc/ssh/ssh\_config\n",  
               "sed -i s/requiretty/\\!requiretty/ /etc/sudoers\n",  
               "hostnamectl set-hostname control\n",  
  
               "yum install -y epel-release\n",  
               "yum install -y ansible sshpass\n",  
               "yum install -y python-boto python-boto3\n",  
  
               "/sbin/useradd ansible\n",  
               "/bin/echo 'G%6rA^T%' | /bin/passwd ansible --stdin\n",  
               "/bin/echo \"ansible        ALL=(ALL)       NOPASSWD: ALL\" >> /etc/sudoers\n",  
               "sudo -u ansible /bin/mkdir -p /home/ansible/.ssh\n",  
               "sudo -u ansible /bin/ssh-keygen -q -N \"\" -f /home/ansible/.ssh/id\_rsa\n",  
               "sudo -u ansible sshpass -p 'G%6rA^T%' ssh-copy-id -i /home/ansible/.ssh/id\_rsa.pub ansible@localhost\n",  
               "echo \"localhost\" >> /etc/ansible/hosts\n",  
  
               "ACCESS=",  
               {"Ref" : "ansiblekey" },  
               "\n",  
               "SECRET=",  
               {"Fn::GetAtt" : [  
                   "ansiblekey",  
                   "SecretAccessKey"  
                 ]  
               },  
               "\n",  
  
               "echo \"export AWS\_ACCESS\_KEY\_ID='KEYACCESS'\" > /home/ansible/accesstemplate\n",  
               "echo \"export AWS\_SECRET\_ACCESS\_KEY='KEYSECRET'\" > /home/ansible/secrettemplate\n",  
               "echo \"#!/bin/sh\"  >> /home/ansible/keys.sh\n",  
               "sed -e \"s|KEYACCESS|$ACCESS|\" /home/ansible/accesstemplate  >> /home/ansible/keys.sh\n",  
               "sed -e \"s|KEYSECRET|$SECRET|\" /home/ansible/secrettemplate  >> /home/ansible/keys.sh\n",  
               "echo \"export AWS\_REGION='us-east-1'\"  >> /home/ansible/keys.sh\n",  
               "chmod 700 /home/ansible/keys.sh\n",  
               "rm -f /home/ansible/accesstemplate /home/ansible/secrettemplate\n",  
  
               "echo \"AWS\_ACCESS\_KEY\_ID: KEYACCESS\" > /home/ansible/accesstemplate\n",  
               "echo \"AWS\_SECRET\_ACCESS\_KEY: KEYSECRET\" > /home/ansible/secrettemplate\n",  
               "sed -e \"s|KEYACCESS|$ACCESS|\" /home/ansible/accesstemplate  >> /home/ansible/keys.yml\n",  
               "sed -e \"s|KEYSECRET|$SECRET|\" /home/ansible/secrettemplate  >> /home/ansible/keys.yml\n",  
               "echo \"AWS\_REGION: us-east-1\"  >> /home/ansible/keys.yml\n",  
               "chmod 600 /home/ansible/keys.yml\n",  
               "rm -f /home/ansible/accesstemplate /home/ansible/secrettemplate\n",  
  
               "chown ansible.ansible /home/ansible/\*\n",  
               "echo \"finished\" > /home/cloud\_user/.done\n"  
           ]]}},  
           "NetworkInterfaces" : [{  
               "GroupSet"                 : [{ "Ref" : "EC2SecurityGroup" }],  
               "AssociatePublicIpAddress": "true",  
               "DeviceIndex"              : "0",  
               "DeleteOnTermination"      : "true",  
               "SubnetId"                 : { "Ref" : "PublicSubnet1" }  
           }]  
       }  
   }  
  },  
 "Outputs" : {  
    "pubIpAddress1" : {  
       "Description" : "Public IP of Ansible Control Node",  
       "Value" : { "Fn::GetAtt" : [ "Control", "PublicIp" ] }  
     },  
     "privIpAddress3" : {  
       "Description" : "Private IP of Ansible Control Node",  
       "Value" : { "Fn::GetAtt" : [ "Control", "PrivateIp" ] }  
     }  
 }  
}

17:

## Combine Git and Ansible to Manage Website Files



# Combine Git and Ansible to Manage Website Files

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## Adam Clegane loves cats. He loves them so much he's decided to have a bunch of pictures taken of them to publish on a website. Now that the website code has been written, it's time to deploy his masterpiece. The website, along with some irrelevant files, are located in this [Git repository](https://github.com/linuxacademy/content-ex447-ansible-best-practice).

You've been contracted to deploy the website to a newly provisioned server. The customer already has an Ansible host set up for your use, but there's no software to serve a web page installed. You'll have to use Ansible to install a web server and use git to pull down the website repository.

## Solution

## Log in to the server using the credentials provided:

ssh cloud\_user@<PUBLIC\_IP\_ADDRESS>

Make sure to elevate to root once logged in.

**Note:** *When copying and pasting code into Vim from the lab guide, first enter :set paste (and then i to enter insert mode) to avoid adding unnecessary spaces and hashes. To save and quit the file, press* ***Escape*** *followed by :wq. To exit the file without saving, press* ***Escape*** *followed by :q!.*

### Write a Playbook to Install git and httpd on the Web Server

1. Elevate to root:

sudo -i

1. Clear your screen:

clear

1. List files and directories:

ls

1. Edit vim install\_software.yml:

vim install\_software.yml

1. Set the following values for each playbook task:

 - hosts: all  
   become: yes  
  
   tasks:  
     - name: Install required software  
       yum:  
         state: present  
         name:  
          - httpd  
          - git

1. To save and exit the file, press **Escape**, type :wq, and hit **Enter**.
2. Run this command:

ansible-playbook -i inventory install\_software.yml

1. SSH into webserver1:

ssh cloud\_user@webserver1

1. Elevate to root:

sudo -i

1. Verify that httpd is installed:

rpm -q httpd

1. Verify that git is installed:

rpm -q git

1. Clear your screen:

clear

### Clone the Provided Repo and Provision the Website Files

1. Create the work directory:

mkdir work

1. Move into the work directory:

cd work/

1. Clone the [GitHub repository](https://github.com/linuxacademy/content-ex447-ansible-best-practice):

git clone <GIT\_REPOSITORY\_LINK/>

1. List the files:

ls

1. Go into the directory:

cd <DIRECTORY\_NAME>

1. View the directories:

ls

1. Move into website:

cd website/

1. Copy the file:

cp -a \* /var/www/html

1. Move to that file:

cd /var/www/html

1. View the directories:

ls

1. Start the unit:

systemctl start httpd

1. Transfer data:

curl localhost

1. Navigate to a web browser and enter the public IP address of Webserver1.

## Conclusion

## Congratulations — you've completed this hands-on lab!

## Additional Resources

There are 2 tasks to accomplish for this hands-on lab.

1. Install both git and httpd on the "WebServer1" server using Ansible.

**Note:** We will be using the term httpd and "Apache" interchangeably throughout this lab.

1. Clone the provided git repository and deploy the website stored there. Verify the deployment worked by visiting the website.

The website is located in this [GitHub link](https://github.com/linuxacademy/content-ex447-ansible-best-practice).

## Learning Objectives

0 of 2 completed

Write a Playbook to Install 'git' and 'httpd' on the Web Server

The provisioned web server currently doesn't have either git or httpd installed. To resolve this, we need to use an Ansible playbook to install the required software.

**Note:** We will be using the term httpd and "Apache" interchangeably throughout this lab.

Clone the Provided Repo and Provision the Website Files

Now that git is installed, we can clone the provided Git repository and put the files where they need to go. Finally, we can verify the website loads correctly.

HOST  :

[root@Server1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash -xe

/usr/bin/hostnamectl set-hostname Server1

/usr/bin/yum install wget -y

dnf install <https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm> -y

dnf install sshpass -y

/usr/bin/echo 'cloud\_user:D=KMc3vJ' | /usr/sbin/chpasswd

/bin/echo 'source /root/ansible/hacking/env-setup -q' >> /root/.bashrc

/bin/echo 'python\_interpreter=auto' >> /etc/ansible/ansible.cfg

/bin/echo 'host\_key\_checking = False' >> /etc/ansible/ansible.cfg

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/bin/echo '[webservers]' >> /root/inventory

/bin/echo 'webserver1 ansible\_user=cloud\_user' >> /root/inventory

/bin/ssh-keygen -t rsa -q -f '/root/.ssh/id\_rsa' -N ''

SSHPASS='D=KMc3vJ' /bin/rsync —rsh='sshpass -e ssh -o StrictHostKeyChecking=no -l cloud\_user' /root/.ssh/id\_rsa.pub 10.0.0.41:/tmp

SSHPASS='D=KMc3vJ' /bin/sshpass -e ssh -o StrictHostKeyChecking=no cloud\_user@webserver1 'cat /tmp/id\_rsa.pub >> /home/cloud\_user/.ssh/authorized\_keys'

/bin/echo 'cloud\_user ALL=(ALL) NOPASSWD:ALL' >> /etc/sudoers.d/90-cloud-init-users

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-3403-fd5fb9152fcb8ad9f6073487d42ebb1a —resource PublicInstance —region us-east-1

[root@Server1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.7 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.7"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.7 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/red_hat_enterprise_linux/8/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.7

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.7"

[root@Server1 cloud\_user]# ansible —version

ansible 2.10.0.dev0

config file = /etc/ansible/ansible.cfg

configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']

ansible python module location = /root/ansible/lib/ansible

executable location = /root/ansible/bin/ansible

python version = 3.6.8 (default, Jan 11 2023, 08:43:50) [GCC 8.5.0 20210514 (Red Hat 8.5.0-16)]

[root@Server1 cloud\_user]#

Webserver 1 :

[root@WebServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash -xe

/usr/bin/hostnamectl set-hostname WebServer1

/usr/bin/yum install wget -y

/sbin/useradd ansible

/usr/bin/echo 'ansible:D=KMc3vJ' | /usr/sbin/chpasswd

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/echo 'cloud\_user:D=KMc3vJ' | /usr/sbin/chpasswd

/sbin/setenforce 0

/bin/echo 'cloud\_user ALL=(ALL) NOPASSWD:ALL' >> /etc/sudoers.d/90-cloud-init-users

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-3403-fd5fb9152fcb8ad9f6073487d42ebb1a —resource PublicInstance2 —region us-east-1

[root@WebServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.9 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

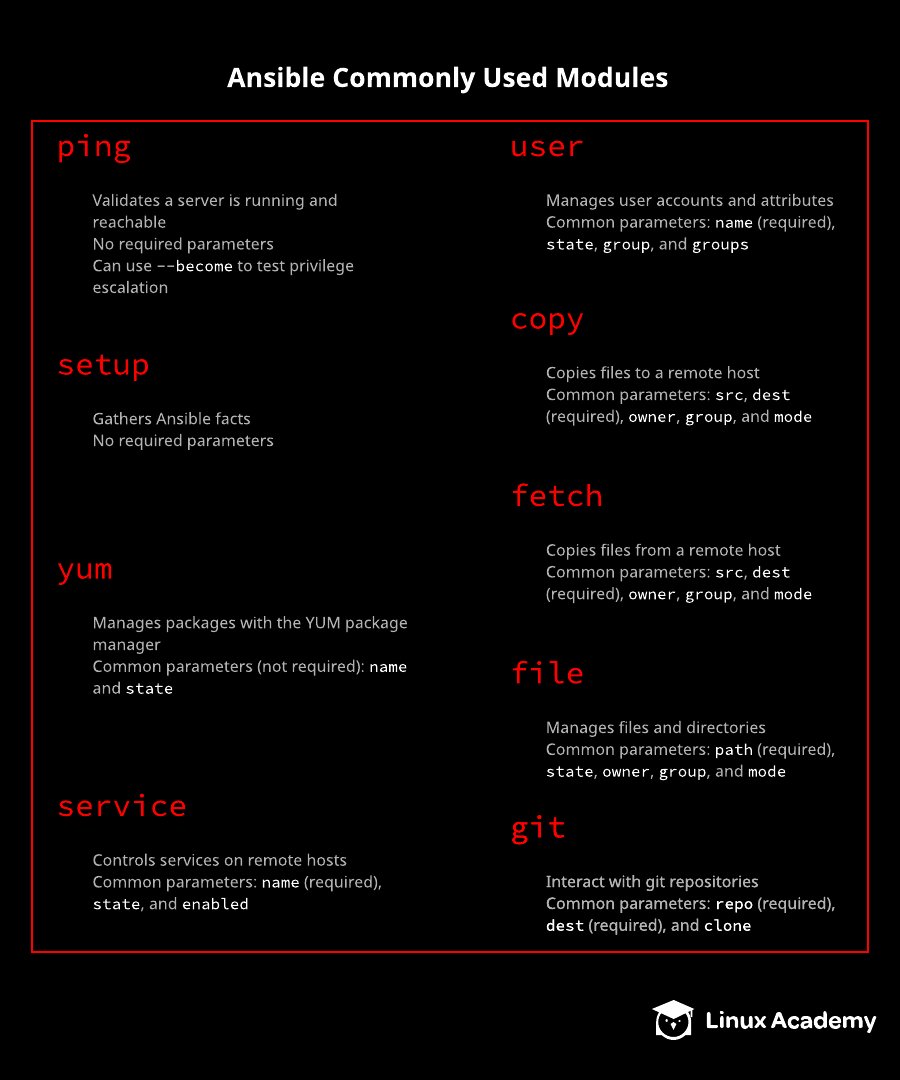
REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@WebServer1 cloud\_user]# ansible —version

bash: ansible: command not found...

18

## Ansible Commonly Used Modules



# Ansible Commonly Used Modules

# *This course is not approved or sponsored by Red Hat.*

## Introduction

## Ansible has many built in modules, but some are used a lot more than others. In this lab we're going to reinforce the usage of some of the more common ones, so that they're easier for you to remember when the time comes.

## Instructions

## Our Ansible environment is set up, so let's take it for a real spin and show our co-workers this proof of concept is worth the time it took to build.

After you verify connectivity in your environment, write three playbooks: one for webservers, one for dbservers, and one for adminservers. Rather than calling for server names individually, use the group names associated with them in /etc/ansible/hosts. Make sure that between all three playbooks, the following gets done:

1. Install httpd on any clients in the webserver group.
2. Start the httpd service on those same clients.
3. Create a dba account and add it to any dbserver group clients.
4. Copy /root/DBAstuff.txt to the new user's home directory. Make sure that user account (and group with the same name) owns the file, and set permissions to **0600**.
5. Create index.html in /var/www/html on webserver group clients.
6. Use the git module to check out <https://github.com/ansible/ansible.git> on adminserver members, and put it in the /opt directory.

## Logging In

## Use the credentials provided on the hands-on lab page to get into Server1 to begin with. Since we need root privileges, let's just run sudo -i right off and become root.

## Verify Connectivity in the Environment

## On the command line run ansible -m ping all to see what servers are out there listening. We should see three groups, each with one server in it.

## Install httpd

## We've got to create a playbook that installs httpd on webserver hosts (vim webservers.yml is one way to do it). When it's finished, the playbook should look something like this:

---  
# Common Modules Playbook  
#  
  
- name: Common Modules Playbook  
  become: yes  
  hosts: webservers  
  
  tasks:  
   - name: Install httpd on webservers  
     yum:  
      name: httpd  
      state: present

Run the playbook, if you want:

ansible-playbook webservers.yml

It's also fine to leave running it until the end. A playbook run only makes changes if something is not in the desired stated. So, if httpd is not installed, this last run would have done it. If it *was* already installed, nothing would have happened at all.

## Start and Enable the httpd Service

## On the same hosts that we want httpd installed, we want it started up too. In this task all we have to do is add the following to the playbook:

   - name: Start the httpd service  
     service:  
      name: httpd  
      state: started  
      enabled: yes

Run the playbook:

ansible-playbook webservers.yml

## Add a dba User Account to dbservers

## Now we've got to make a second playbook, the one that will add a dba account to clients in the dbserver group. Create it (with something like vim dbserver.yml) and it should look similar to this when we're done:

---  
  
- name: DB server playbook  
  become: yes  
  hosts: dbservers  
  
  tasks:  
   - name: Add the 'dba' user to the dbservers  
     user:  
       name: dba

Run the playbook, if you like:

ansible-playbook dbserver.yml

## Copy the Required File to Clients in dbservers

## We have to get /root/DBAstuff.txt to this new user's home directory. If we add something like this to our dbserver playbook, we should be good to go:

   - name: Copy required DBA files  
     copy:  
      src: /root/DBAstuff.txt  
      dest: /home/dba/  
      owner: dba  
      group: dba  
      mode: 0600

Run the playbook:

ansible-playbook dbserver.yml

## Create /var/www/html/index.html

## We need to have an index.html created on any Ansible client in the the webserver group. This means we've got to tack a bit onto the end of our webserver playbook. That extra piece should look like this:

   - name: Create index.html  
     file:  
      path: /var/www/html/index.html  
      state: touch

Run the playbook again:

ansible-playbook webservers.yml

## Clone the Ansible Git Repository into /opt for adminservers Members

## We want to clone the Ansigle Git repo into the /opt directory on any Ansible client that's a member of the adminserver group. To do that, we're going to create another playbook (using vim admins.yml as an example) and make it look like this:

---  
  
- name: Use git to clone the Ansible repo  
  hosts: admins  
  become: yes  
  
  tasks:  
   - name: Use the git module  
     git:  
      repo: https://github.com/ansible/ansible.git  
      dest: /opt

If we try running this one, using ansible-playbook admins.yml, the run will fail. We don't have git installed. So let's go back and fix that in the playbook. We'll put a little "Install git" task before the "Use the git module" task. It should look like this:

---  
  
- name: Use git to clone the Ansible repo  
  hosts: admins  
  become: yes  
  
  tasks:  
   - name: Install git  
     yum:  
      name: git  
      state: present  
  
   - name: Use the git module  
     git:  
      repo: https://github.com/ansible/ansible.git  
      dest: /opt

Now try running it again, and it should work.

## Conclusion

## And that actually marks the end of this hands-on lab. We installed httpd and got it started on any clients in the webserver group, then added a user account to any hosts in the dbserver group and made sure there was a text file that got copied to that user's home directory. After that we created a default file that httpd on those webserver group members will serve out. And finally, on any clients in the adminserver group, we checked out Ansible from a Git repository and stuck in in /opt. That's a fair bit of heavy lifting that Ansible is doing for us now. Congratulations!

## Additional Resources

Notice: Ansible is installed as the root user, so please work on all tasks after elevating to the root user.

Your Ansible environment is set up so let's take it for a real spin so you can show your co-workers this proof of concept is worth the time.

After you verify connectivity in your environment, write playbooks that do the following - 1 for webservers, 1 for dbservers, and 1 for admin servers. Rather than calling for server names individually, use the group names associated with them in /etc/ansible/hosts.

1. Install httpd on the webserver group.
2. Start the httpd service on the webserver group.
3. Create a dba account on the dbserver group.
4. Copy /root/DBAstuff.txt to the new user's home directory. Make sure he is the owner and group of the file, set permissions to 0600.
5. Create index.html in /var/www/html on the webserver.
6. Use the git module to check out <https://github.com/ansible/ansible.git> on the Admin server - put it in the /opt directory.

## Learning Objectives

0 of 7 completed

Verify Connectivity in the Environment

On the command line run ansible -m ping all.

Install httpd

Your playbook, webservers.yml should look something like this:

---  
# Common Modules Playbook  
#  
  
- name: Common Modules Playbook  
  become: yes  
  hosts: webservers  
  
  tasks:  
   - name: Install httpd on webservers  
     yum:  
      name: httpd  
      state: present

Start and Enable the httpd Service

In this task all we have to do is add the following to the webservers.yml playbook:

   - name: Start the httpd service  
     service:  
      name: httpd  
      state: started  
      enabled: yes

Create a dba User Account

Your playbook, dbserver.yml, should look similar to this:

---  
  
- name: DB server playbook  
  become: yes  
  hosts: dbservers  
  
  tasks:  
   - name: Add the 'dba' user to the dbservers  
     user:  
       name: dba

Copy the Required File

You can add to your dbserver.yml playbook something similar to the following:

   - name: Copy required DBA files  
     copy:  
      src: /root/DBAstuff.txt  
      dest: /home/dba/  
      owner: dba  
      group: dba  
      mode: 0600

Create /var/www/html/index.html

The final addition to your webserver.yml playbook should look similar to:

   - name: Create index.html  
     file:  
      path: /var/www/html/index.html  
      state: touch

Clone the Ansible Git Repository into /opt on adminservers

Your admin.yml playbook will look something like this:

---  
  
- name: Use git to clone the Ansible repo  
  hosts: admins  
  become: yes  
  
  tasks:  
   - name: Install git  
     yum:  
      name: git  
      state: present  
  
   - name: Use the git module  
     git:  
      repo: https://github.com/ansible/ansible.git  
      dest: /opt

Ansible Host :

[root@Server1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname Server1

/usr/bin/yum install wget git -y

/bin/echo 'source /root/ansible/hacking/env-setup -q' >> /root/.bashrc

/bin/ssh-keygen -N '' -t rsa -f ~/.ssh/id\_rsa

/bin/cat /root/.ssh/id\_rsa.pub >> /home/ansible/.ssh/authorized\_keys

/bin/echo 'python\_interpreter=auto' >> /etc/ansible/ansible.cfg

/bin/echo 'host\_key\_checking = False' >> /etc/ansible/ansible.cfg

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/DBAstuff.txt> -O /root/DBAstuff.txt

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/privkey> -O /tmp/pkey

/bin/cat /tmp/pkey > /root/.ssh/id\_rsa

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/akey

/bin/cat /tmp/akey > /root/.ssh/id\_rsa.pub

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:Dcf8RA[Q' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2266-afdec2a3c848c7916118e2cb2df9172c —resource PublicInstance —region us-east-1

[root@Server1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.7 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.7"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.7 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/red_hat_enterprise_linux/8/>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.7

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.7"

[root@Server1 cloud\_user]# ansible —version

ansible 2.10.0.dev0

config file = /etc/ansible/ansible.cfg

configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']

ansible python module location = /root/ansible/lib/ansible

executable location = /root/ansible/bin/ansible

python version = 3.6.8 (default, Jan 11 2023, 08:43:50) [GCC 8.5.0 20210514 (Red Hat 8.5.0-16)]

[root@Server1 cloud\_user]#

Webserver 1 :

[root@WebServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname WebServer1

/usr/bin/yum install wget -y

/sbin/useradd ansible

/usr/bin/echo 'ansible:Dcf8RA[Q' | /usr/sbin/chpasswd

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/key

/bin/mkdir /home/ansible/.ssh

/usr/bin/chmod 0700 /home/ansible/.ssh/

/bin/cat /tmp/key > /home/ansible/.ssh/authorized\_keys

/usr/bin/chmod 0600 /home/ansible/.ssh/authorized\_keys

/usr/bin/chown ansible.ansible /home/ansible -R

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:Dcf8RA[Q' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2266-afdec2a3c848c7916118e2cb2df9172c —resource PublicInstance2 —region us-east-1

[root@WebServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.9 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@WebServer1 cloud\_user]# ansible —version

bash: ansible: command not found...

DB SERVER :

[root@DBServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname DBServer1

/usr/bin/yum install wget -y

/sbin/useradd ansible

/usr/bin/echo 'ansible:Dcf8RA[Q' | /usr/sbin/chpasswd

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/key

/bin/mkdir /home/ansible/.ssh

/usr/bin/chmod 0700 /home/ansible/.ssh/

/bin/cat /tmp/key > /home/ansible/.ssh/authorized\_keys

/usr/bin/chmod 0600 /home/ansible/.ssh/authorized\_keys

/usr/bin/chown ansible.ansible /home/ansible -R

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:Dcf8RA[Q' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2266-afdec2a3c848c7916118e2cb2df9172c —resource PublicInstance3 —region us-east-1

[root@DBServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.9 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@DBServer1 cloud\_user]# ansible —version

bash: ansible: command not found...

Admin server :

[root@AdminServer1 cloud\_user]# curl 169.254.169.254/latest/user-data

#!/bin/bash

/usr/bin/hostnamectl set-hostname AdminServer1

/usr/bin/yum install wget -y

/sbin/useradd ansible

/usr/bin/echo 'ansible:Dcf8RA[Q' | /usr/sbin/chpasswd

/sbin/alternatives —set python /usr/bin/python3

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/etchosts> -O /tmp/hosts

/bin/cat /tmp/hosts > /etc/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/inventory> -O /tmp/inv

/bin/cat /tmp/inv > /etc/ansible/hosts

/usr/bin/wget <https://raw.githubusercontent.com/linuxacademy/content-linux-rhcev8/master/authkey> -O /tmp/key

/bin/mkdir /home/ansible/.ssh

/usr/bin/chmod 0700 /home/ansible/.ssh/

/bin/cat /tmp/key > /home/ansible/.ssh/authorized\_keys

/usr/bin/chmod 0600 /home/ansible/.ssh/authorized\_keys

/usr/bin/chown ansible.ansible /home/ansible -R

/bin/echo 'ansible ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers

/usr/bin/echo 'cloud\_user:Dcf8RA[Q' | /usr/sbin/chpasswd

/usr/bin/pip3 install <https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-py3-latest.tar.gz>

/usr/local/bin/cfn-signal -e 0 —stack cfst-2266-afdec2a3c848c7916118e2cb2df9172c —resource PublicInstance4 —region us-east-1

[root@AdminServer1 cloud\_user]# cat /etc/os-release

NAME="Red Hat Enterprise Linux"

VERSION="8.9 (Ootpa)"

ID="rhel"

ID\_LIKE="fedora"

VERSION\_ID="8.9"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="Red Hat Enterprise Linux 8.9 (Ootpa)"

ANSI\_COLOR="0;31"

CPE\_NAME="cpe:/o:redhat:enterprise\_linux:8::baseos"

HOME\_URL="<https://www.redhat.com/>“

DOCUMENTATION\_URL="<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8>“

BUG\_REPORT\_URL="<https://bugzilla.redhat.com/>“

REDHAT\_BUGZILLA\_PRODUCT="Red Hat Enterprise Linux 8"

REDHAT\_BUGZILLA\_PRODUCT\_VERSION=8.9

REDHAT\_SUPPORT\_PRODUCT="Red Hat Enterprise Linux"

REDHAT\_SUPPORT\_PRODUCT\_VERSION="8.9"

[root@AdminServer1 cloud\_user]# ansible —version

bash: ansible: command not found...