***Module 2***

***Questions***

[user@server4 ~]$ ssh-keygen

Generating public/private rsa key pair.

Enter file in which to save the key (/home/user/.ssh/id\_rsa):

Created directory '/home/user/.ssh'.

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /home/user/.ssh/id\_rsa.

Your public key has been saved in /home/user/.ssh/id\_rsa.pub.

The key fingerprint is:

b8:49:0a:1f:71:f9:0c:54:c8:8e:30:9b:93:b3:3c:bf user@server4.mylabserver.com

The key's randomart image is:

+--[ RSA 2048]----+

|     ..o.        |

|  o  .o.         |

|   \*.o+          |

|  \* .o.=         |

| ..+. o S        |

|  +o + o         |

|   oo o          |

|    .            |

|    E.           |

+-----------------+

[user@server4 ~]$ cd .ssh

[user@server4 .ssh]$ ll

total 8

-rw-------. 1 user user 1679 Sep 21 14:23 id\_rsa

-rw-r--r--. 1 user user  408 Sep 21 14:23 id\_rsa.pub

*Exchange the public key with 'Server2' and the 'user' account on that server. Verify that the identity appearing in the appropriate key file matches the user and system name/IP of the originating 'Server1'.*

[user@server4 .ssh]$ ssh-copy-id 54.86.179.231

The authenticity of host '54.86.179.231 (54.86.179.231)' can't be established.

ECDSA key fingerprint is 0d:0c:b1:1d:e1:cf:6d:9f:51:bf:0f:dc:60:82:a1:73.

Are you sure you want to continue connecting (yes/no)? yes

/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed

/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys

user@54.86.179.231's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '54.86.179.231'"

and check to make sure that only the key(s) you wanted were added.

[user@server4 .ssh]$ ssh 54.86.179.231

Last login: Mon Sep 21 14:22:42 2015 from 216.46.60.98

[user@server5 ~]$ cd .ssh

[user@server5 .ssh]$ ll

total 4

-rw-------. 1 user user 408 Sep 21 14:24 authorized\_keys

[user@server5 .ssh]$ cat authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDBk5qaPck8F+aHk1BZBNmgqW6TSXctE0N+MddROuqm2TvxMeI4s5bfCpWXSBKqhTo/SFqMjM7nHp4F8stG86VEP5UymiWaeVu6rrzxTrT/9QIlFW+yWv9Mbg/GNN48rqDdZ4931sLsYnKx/dDh1S/iTofQpzgl6LB+bpsgWanaVPIcKJkh0LP9LZXX0+VpPlMrk9Gpn1F0k1EdO5zPYyEI2MkmhJZCMKhz3dOZCIqMuayt1QKSSpyDeVDx3dMMuFddXJOyuWEIYtIJC+u7hIIHV6Bsmzl4PI0dHnvmHk+Sn2DBEIn/K+9aqRQMT9IR2a2TOX1EWBqaNDsxIeLjdhw3 user@server4.mylabserver.com

*Verify that you are able to log into 'Server2' from 'Server1' with the 'user' account without entering a password. Check the hostname file to be sure you are logged into 'Server2'.*

[user@server4 .ssh]$ ssh 54.86.179.231

Last login: Mon Sep 21 14:22:42 2015 from 216.46.60.98

[user@server5 .ssh]$ hostname

server5.mylabserver.com

*Repeat Step #1 for 'Server2' (creating and exchanging the 'user' SSH keys with 'Server1'). Create a file on 'Server1' using any method you choose. Use 'secure copy' to copy that file to 'Server2'. Confirm the copy succeeds without password entry.*

[user@server4 ~]$ echo "test file" > testfile.txt

[user@server4 ~]$ scp testfile.txt 54.86.179.231:/home/user

testfile.txt

*Using 'secure copy', copy the file from Step #4 back to 'Server1' from 'Server2', renaming it to something else during the copy process, verify that no password was required.*

[user@server4 ~]$ scp 54.86.179.231:/home/user/testfile.txt copiedfrom.txt

testfile.txt                                                                                                                                                                   100%   10     0.0KB/s   00:00

[user@server4 ~]$ ll

total 8

-rw-rw-r--. 1 user user  10 Sep 21 14:31 copiedfrom.txt

drwxr-xr-x. 2 user user   6 Jan  7  2015 Desktop

-rw-rw-r--. 1 user user   0 Sep 21 14:30 testfile.txt

*Verify that Ansible is running by displaying the version and module path. Change to the Ansible configuration directory and list the contents.*

[root@server3 ~]# ansible --version

ansible 1.9.2

  configured module search path = None

[root@server3 ~]# cd /etc/ansible

[root@server3 ansible]# ll

total 20

-rw-r--r--. 1 root root 8629 Jun 25 21:11 ansible.cfg

-rw-r--r--. 1 root root   88 Sep 21 14:43 hosts

drwxr-xr-x. 2 root root    6 Jun 25 21:11 roles

*Move the original Ansible Hosts file to another file in the same directory called 'hosts.original'. Create a new empty 'hosts' file in the default configuration directory location.*

[root@server3 ansible]# mv hosts hosts.original && touch hosts && ll

total 20

-rw-r--r--. 1 root root 8629 Jun 25 21:11 ansible.cfg

-rw-r--r--. 1 root root   0 Sep 21 14:43 hosts

-rw-r--r--. 1 root root  965 Jun 25 21:11 hosts.original

drwxr-xr-x. 2 root root    6 Jun 25 21:11 roles

*Create a section in the new 'hosts' file called 'local'. Make sure it contains:*

  \* Localhost

  \* Localhost.Localdomain

  \* 127.0.0.1

[root@server3 ansible]# vim hosts

[root@server3 ansible]# cat hosts

[local]

127.0.0.1

localhost

localhost.localdomain

*Create a second section called 'web hosts' with the NAME of the second Edurekalab server in your environment (see previous lab for setting up the two host environment). Display the contents of the file.*

[root@server3 ansible]# vim hosts

[root@server3 ansible]# cat hosts

[local]

127.0.0.1

localhost

localhost.localdomain

[web hosts]

server4.mylabserver.com

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'setup' module for ansible, list all of the known facts on the local system.*

[test@server3 ~]$ ansible local -m setup

localhost | success >> {

    "ansible\_facts": {

        "ansible\_all\_ipv4\_addresses": [

            "172.31.107.208"

        ],

        "ansible\_all\_ipv6\_addresses": [

            "fe80::103f:68ff:fefe:2bd1"

        ],

        "ansible\_architecture": "x86\_64",

        "ansible\_bios\_date": "05/06/2015",

        "ansible\_bios\_version": "4.2.amazon",

        "ansible\_cmdline": {

            "BOOT\_IMAGE": "/boot/vmlinuz-3.10.0-229.14.1.el7.x86\_64",

        (NOTE: Your output will be larger as this was concatenated for space and readability)

*Using the 'setup' module for ansible, list all of the known facts for all systems configured in the 'hosts' file on the system.*

[test@server3 ~]$ ansible all -m setup

(NOTE: Localhost truncated here... starting with another host output)

server4.mylabserver.com | success >> {

    "ansible\_facts": {

        "ansible\_all\_ipv4\_addresses": [

            "172.31.108.45"

        ],

        "ansible\_all\_ipv6\_addresses": [

            "fe80::1029:fbff:fed6:9aaf"

        ],

        "ansible\_architecture": "x86\_64",

        "ansible\_bios\_date": "05/06/2015",

        "ansible\_bios\_version": "4.2.amazon",

        "ansible\_cmdline": {

            "BOOT\_IMAGE": "/boot/vmlinuz-3.10.0-229.14.1.el7.x86\_64",

   (NOTE: Your output will be larger as this was concatenated for space and readability)

*While listing the facts on the local system, filter the list showing only the content pertaining to the system IP addresses.*

[test@server3 ~]$ ansible local -m setup -a 'filter=ans\*ipv4\*'

localhost | success >> {

    "ansible\_facts": {

        "ansible\_all\_ipv4\_addresses": [

            "172.31.107.208"

        ],

        "ansible\_default\_ipv4": {

            "address": "172.31.107.208",

            "alias": "eth0",

            "gateway": "172.31.96.1",

            "interface": "eth0",

            "macaddress": "12:3f:68:fe:2b:d1",

            "mtu": 9001,

            "netmask": "255.255.240.0",

            "network": "172.31.96.0",

            "type": "ether"

        }

    },

    "changed": false

}

*Module 3*

NOTE: These exercises can be completed on Edureka Lab Servers with any distribution and version available AS LONG AS the Ansible application has been installed and the 'test' user has been set up with SSH Key Exchange for password less SSH sessions (see previous lab for setup).

***Questions***

*Display the current version and status of the Ansible application.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Using the appropriate module, from your control server, list all the hosts configured in your environment. Using ansible, run a ping command against all hosts in the environment.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

[test@server3 ~]$ ansible all -m ping

localhost | success >> {

    "changed": false,

    "ping": "pong"

}

server5.mylabserver.com | success >> {

    "changed": false,

    "ping": "pong"

}

server4.mylabserver.com | success >> {

    "changed": false,

    "ping": "pong"

}

*List the contents of your ansible 'hosts' file. Using only ONE of the groups in the file, attempt to install the 'lynx' package on that server.*

[test@server3 ~]$ cat /etc/ansible/hosts

[local]

localhost

[apacheweb]

server4.mylabserver.com

[appserver]

server5.mylabserver.com

[test@server3 ~]$ ansible apacheweb -s -m shell -a 'yum -y install lynx'

server4.mylabserver.com | success | rc=0 >>

Loaded plugins: fastestmirror

Loading mirror speeds from cached hostfile

 \* base: distro.ibiblio.org

 \* epel: mirror.symnds.com

 \* extras: mirror.cogentco.com

 \* updates: mirror.cogentco.com

Package lynx-2.8.8-0.3.dev15.el7.x86\_64 already installed and latest version

Nothing to do

*Using the same group as Step #3, attempt to install the 'telnet' package on that server using the more 'playbook friendly' module and syntax.*

[test@server3 ~]$ ansible apacheweb -s -m yum -a 'pkg=telnet state=installed update\_cache=true'

server4.mylabserver.com | success >> {

    "changed": false,

    "msg": "",

    "rc": 0,

    "results": [

        "telnet-0.17-59.el7.x86\_64 providing telnet is already installed"

    ]

}

*On the control server, display the ansible version and status and issue an ansible command to list all configured hosts.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Choosing ONE of the groups from the displayed list in Step #1 above, query that system for all the 'facts' that can be displayed, while filtering the content for the IP address information*.

[test@server3 ~]$ ansible apacheweb -m setup -a 'filter=ans\*ipv4\*'

server4.mylabserver.com | success >> {

    "ansible\_facts": {

        "ansible\_all\_ipv4\_addresses": [

            "172.31.108.45"

        ],

        "ansible\_default\_ipv4": {

            "address": "172.31.108.45",

            "alias": "eth0",

            "gateway": "172.31.96.1",

            "interface": "eth0",

            "macaddress": "12:29:fb:d6:9a:af",

            "mtu": 9001,

            "netmask": "255.255.240.0",

            "network": "172.31.96.0",

            "type": "ether"

        }

    },

    "changed": false

}

*Using the same group from Steps #1 and #2, issue a shell command through ansible that will determine if the 'lynx' package is already installed on that server.*

[test@server3 ~]$ ansible apacheweb -m shell -a 'yum list installed | grep lynx'

server4.mylabserver.com | success | rc=0 >>

lynx.x86\_64                           2.8.8-0.3.dev15.el7              @base

*Using the same group, issue an ansible command (as sudo) that will display the last ten lines of output from the remote system's 'syslog' file.*

[test@server3 ~]$ ansible apacheweb -m shell -a 'tail -n 10 /var/log/dmesg'

server4.mylabserver.com | success | rc=0 >>

[    3.577455] SELinux: initialized (dev autofs, type autofs), uses genfs\_contexts

[    3.621406] SELinux: initialized (dev hugetlbfs, type hugetlbfs), uses transition SIDs

[    3.748973] systemd-udevd[504]: starting version 208

[    3.828536] Installing knfsd (copyright (C) 1996 okir@monad.swb.de).

[    3.839265] SELinux: initialized (dev nfsd, type nfsd), uses genfs\_contexts

[    3.949670] piix4\_smbus 0000:00:01.3: SMBus base address uninitialized - upgrade BIOS or use force\_addr=0xaddr

[    4.082323] input: PC Speaker as /devices/platform/pcspkr/input/input4

[    4.191443] type=1305 audit(1443204264.317:4): audit\_pid=546 old=0 auid=4294967295 ses=4294967295 subj=system\_u:system\_r:auditd\_t:s0 res=1

[    4.277258] ppdev: user-space parallel port driver

[    4.459507] Adding 2097148k swap on /root/swap.  Priority:-1 extents:2 across:3037200k SSFS

While logged into your control server, create a directory called 'Playbooks' and create an empty file in it called 'deploy\_DATE.yml'. Replace DATE with today's date.

[test@server3 ~]$ mkdir Playbooks

[test@server3 ~]$ cd Playbooks

[test@server3 ~]$ touch deploy\_09242015.yml

*Edit the 'deploy\_DATE.yml' file and create a structure that will run the following against ONE of the groups in your host configuration:*

*- Using the package installation module, install lynx package*

*- Using the package installation module, determine if the telnet package is installed*

CONTENT SHOULD LOOK SOMETHING LIKE:

- hosts: appserver

  tasks:

  - name: Install Lynx on App Servers

    yum: pkg=lynx state=installed update\_cache=true

  - name: Querying for Telnet Install

    yum: pkg=telnet state=present update\_cache=true

Run the playbook and display the results.

[test@server3 Playbooks]$ ansible-playbook -s deploy\_09242015.yml

PLAY [appserver] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server5.mylabserver.com]

 TASK: [Install Lynx on App Servers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server5.mylabserver.com]

 TASK: [Querying for Telnet Install] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server5.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server5.mylabserver.com      : ok=3    changed=0    unreachable=0    failed=0

*Edit the playbook in Step #2 and create a new section for a DIFFERENT group in your host configuration as follows:*

*- Using the package installation module, install the telnet package*

*- Using the package installation module, determine if the lynx package is installed*

FULL FILE SHOULD NOW LOOK SOMETHING LIKE:

- hosts: appserver

  tasks:

  - name: Install Lynx on App Servers

    yum: pkg=lynx state=installed update\_cache=true

  - name: Querying for Telnet Install

    yum: pkg=telnet state=present update\_cache=true

- hosts: apacheweb

  tasks:

  - name: Install Lynx on Web Servers

    yum: pkg=telnet state=installed update\_cache=true

  - name: Querying for Lynx Install

    yum: pkg=lynx state=present update\_cache=true

Run the full playbook and display the results.

[test@server3 Playbooks]$ ansible-playbook -s deploy\_09242015.yml

 PLAY [appserver] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server5.mylabserver.com]

 TASK: [Install Lynx on App Servers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server5.mylabserver.com]

 TASK: [Querying for Telnet Install] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server5.mylabserver.com]

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [Install Lynx on Web Servers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [Querying for Lynx Install] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=3    changed=0    unreachable=0    failed=0

server5.mylabserver.com      : ok=3    changed=0    unreachable=0    failed=0

*Tail the last lines of the ansible log file and compare to the results in Step #5.*

[test@server3 Playbooks]$ tail /var/log/ansible.log

2015-09-25 18:25:43,538 p=1682 u=test |

2015-09-25 18:25:43,569 p=1682 u=test |  PLAY [appserver] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2015-09-25 18:25:43,569 p=1682 u=test |  GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2015-09-25 18:25:43,782 p=1682 u=test |  ok: [server5.mylabserver.com]

2015-09-25 18:25:43,783 p=1682 u=test |  TASK: [Install Lynx on App Servers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2015-09-25 18:25:47,166 p=1682 u=test |  ok: [server5.mylabserver.com]

2015-09-25 18:25:47,166 p=1682 u=test |  TASK: [Querying for Telnet Install] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2015-09-25 18:25:51,595 p=1682 u=test |  ok: [server5.mylabserver.com]

2015-09-25 18:25:51,596 p=1682 u=test |  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2015-09-25 18:25:51,596 p=1682 u=test |  server5.mylabserver.com      : ok=3    changed=0    unreachable=0    failed=0

*Run the appropriate ansible command to list all the hosts that are configured on your system for access by the control server. Display the ansible 'hosts' file in active use on the control server so that the hosts and groups are displayed.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

[test@server3 ~]$ cat /etc/ansible/hosts

[local]

localhost

[apacheweb]

server4.mylabserver.com

[appserver]

server5.mylabserver.com

*Create a directory called "Playbooks". Change to that directory and create a playbook called 'myfirstplaybook.yml' in that directory. Edit the file (and then display) so that it has the initial comment at the top (formatted appropriately) called "My First Playbook".*

 [test@server3 ~]$ mkdir Playbooks

[test@server3 ~]$ cd Playbooks

[test@server3 Playbooks]$ vim myfirstplaybook.yml

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

*Edit the playbook and create a target section that will run against either one of the defined hosts in Step #1 above or a group of hosts in Step #1 above. Save the plabook and execute the appropriate command to run the playbook and display the results.*

[test@server3 Playbooks]$ vim myfirstplaybook.yml

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

[test@server3 Playbooks]$ ansible-playbook -s myfirstplaybook.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=1    changed=0    unreachable=0    failed=0

*Edit the playbook and add the following to the target section we just created:*

*- Force use of SSH connections*

*- Always run the playbook as the user 'test'*

*- Run this playbook as SUDO by default*

*- Do not use the setup module to gather facts from systems during execution*

[test@server3 Playbooks]$ vim myfirstplaybook.yml

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

*Display the file and run the playbook, displaying the results of the run.*

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

 [test@server3 Playbooks]$ ansible-playbook myfirstplaybook.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*Run the appropriate ansible command to list all the hosts that are configured on your system for access by the control server. Display the ansible 'hosts' file in active use on the control server so that the hosts and groups are displayed.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

[test@server3 ~]$ cat /etc/ansible/hosts

[local]

localhost

[apacheweb]

server4.mylabserver.com

[appserver]

server5.mylabserver.com

*Create a directory called 'conf'. Change to that directory and create two files, one called 'copyright.yml' and one called 'webdefaults.yml'. In the copyright file, create variables called 'message' and 'author' containing appropriate text of your choice. In the webdefaults file, create two variables called 'apache\_version' and 'apache\_mod\_req' with values of your choice. Display the files.*

[test@server3 Playbooks]$ mkdir conf

[test@server3 Playbooks]$ cd conf

[test@server3 conf]$ cat

copyright.yml    webdefaults.yml

[test@server3 conf]$ cat copyright.yml

---

message: Copyright 2015 by Edureka

author: Edureka

[test@server3 conf]$ cat webdefaults.yml

---

apache\_version: 2.6

apache\_mod\_req: mod\_ssl

*Change back to the 'Playbooks' directory and edit your 'myfirstplaybook.yml' file. Create a variables section that contains a local variable called 'playbook\_version' and set it to a value appropriate to your desires. Display the file and then run the playbook.*

[test@server3 Playbooks]$ vim myfirstplaybook.yml

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  vars:

    playbook\_version: 0.1b

[test@server3 Playbooks]$ ansible-playbook myfirstplaybook.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*Edit the 'myfirstplaybook.yml' file and create another variables section that appropriately imports the external files that we created in Step #2. Display the newly edited file.*

[test@server3 Playbooks]$ vim myfirstplaybook.yml

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  vars:

    playbook\_version: 0.1b

  vars\_files:

    - conf/copyright.yml

    - conf/webdefaults.yml

Run the ansible playbook just edited and note the results.

[test@server3 Playbooks]$ ansible-playbook myfirstplaybook.yml

PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*Run the appropriate ansible command to list all the hosts that are configured on your system for access by the control server. Display the ansible 'hosts' file in active use on the control server so that the hosts and groups are displayed.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

[test@server3 ~]$ cat /etc/ansible/hosts

[local]

localhost

[apacheweb]

server4.mylabserver.com

[appserver]

server5.mylabserver.com

*Edit the previously created 'myfirstplaybook.yml' file from our previous exercises. Create a task section that uses the 'yum' module to install the apache web server on the host(s) indicated in the playbook. Display the file after editing.*

[test@server3 Playbooks]$ vim myfirstplaybook.yml

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  vars:

    playbook\_version: 0.1b

  vars\_files:

    - conf/copyright.yml

    - conf/webdefaults.yml

  tasks:

    - name: Install Apache Web Server

      action: yum name=httpd state=installed

*Run the playbook, after the initial installation of apache, run the playbook a second time to note the differing output.*

[test@server3 Playbooks]$ ansible-playbook myfirstplaybook.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Install Apache Web Server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=1    changed=1    unreachable=0    failed=0

 [test@server3 Playbooks]$ ansible-playbook myfirstplaybook.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK: [Install Apache Web Server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=1    changed=0    unreachable=0    failed=0

*Add another task to the playbook that uses the 'yum' module to check and see if the 'lynx' package is installed and then display that file.*

[test@server3 Playbooks]$ vim myfirstplaybook.yml

[test@server3 Playbooks]$ cat myfirstplaybook.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  vars:

    playbook\_version: 0.1b

  vars\_files:

    - conf/copyright.yml

    - conf/webdefaults.yml

  tasks:

    - name: Install Apache Web Server

      action: yum name=httpd state=installed

    - name: Verify the Lynx Web Browser

      action: yum name=lynx state=present

*Run the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook myfirstplaybook.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Install Apache Web Server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [Verify the Lynx Web Browser] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=2    changed=0    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a playbook, using the concepts discussed in the LOOP video:*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- The playbook runs as 'sudo'*

*- Skip gathering remote facts*

*- Defines a list of users within the playbook, using the 'user' module, iterates through that list of users, adding them all to the remote system*

[test@server3 Playbooks]$ vim loop.yml

[test@server3 Playbooks]$ cat loop.yml

--- # LOOP Playbook Example

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  tasks:

    - name: Add a list of users

      user: name={{ item }} state=present

      with\_items:

        - user1

        - user2

        - user3

*Run the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook loop.yml

PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Add a list of users] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com] => (item=user1)

ok: [server4.mylabserver.com] => (item=user2)

ok: [server4.mylabserver.com] => (item=user3)

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=1    changed=0    unreachable=0    failed=0

 [test@server3 Playbooks]$ ssh server4

Last login: Wed Oct 14 15:22:34 2015 from ec2-52-91-231-138.compute-1.amazonaws.com

[test@server4 ~]$ cat /etc/passwd| grep user

user1:x:1005:1005::/home/user1:/bin/bash

user2:x:1006:1006::/home/user2:/bin/bash

user3:x:1007:1007::/home/user3:/bin/bash

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a playbook, using the 'NOTIFY' concepts from the video:*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- The playbook runs as 'sudo'*

*- Skip gathering remote facts*

*- Installs the 'Apache Web Server' using the appropriate package module*

*- Upon installation of the web server, notifies the appropriately titled handler to restart the service using the 'service' module*

[test@server3 Playbooks]$ vim notify.yml

[test@server3 Playbooks]$ cat notify.yml

--- # My First YAML Playbook for Ansible

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  tasks:

    - name: Install Apache Web Server

      action: yum name=httpd state=installed

      notify: Restart HTTPD

  handlers:

    - name: Restart HTTPD

      action: service name=httpd state=restarted

*Run the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook notify.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Install Apache Web Server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

NOTIFIED: [Restart HTTPD] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=2    changed=2    unreachable=0    failed=0

[test@server3 Playbooks]$ ssh server4

Last login: Wed Oct 14 15:27:43 2015 from ec2-52-91-231-138.compute-1.amazonaws.com

[test@server4 ~]$ ps aux | grep http

root      2042  0.0  0.4 213080  4752 ?        Ss   15:27   0:00 /usr/sbin/httpd -DFOREGROUND

apache    2043  0.0  0.2 213080  2752 ?        S    15:27   0:00 /usr/sbin/httpd -DFOREGROUND

apache    2044  0.0  0.2 213080  2752 ?        S    15:27   0:00 /usr/sbin/httpd -DFOREGROUND

apache    2045  0.0  0.2 213080  2752 ?        S    15:27   0:00 /usr/sbin/httpd -DFOREGROUND

apache    2046  0.0  0.2 213080  2752 ?        S    15:27   0:00 /usr/sbin/httpd -DFOREGROUND

apache    2047  0.0  0.2 213080  2752 ?        S    15:27   0:00 /usr/sbin/httpd -DFOREGROUND

test      2077  0.0  0.0 112640   964 pts/0    R+   15:28   0:00 grep --color=auto http

***Module 4***

***Questions***

*Run the appropriate ansible command to list all the hosts that are configured on your system for access by the control server. Display the ansible 'hosts' file in active use on the control server so that the hosts and groups are displayed.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

[test@server3 ~]$ cat /etc/ansible/hosts

[local]

localhost

[apacheweb]

server4.mylabserver.com

[appserver]

server5.mylabserver.com

*Create a full Ansible Playbook called 'pause.yml' with the following characteristics, incorporating the PAUSE module, display the playbook once created:*

*- Connect to one (or a group) of hosts from the list displayed in Step #1 above*

*- Force the playbook to run with 'sudo' privileges*

*- Skip gathering facts on the remote system(s)*

*- Using the appropriate package management module, install HTTPD/Apache on the remote system(s)*

*- Using the PAUSE Module, Pause the Playbook so that the user has to press enter to continue the run*

*- Using the appropriate package maangement module, check to see if the 'lynx' browser is installed*

[test@server3 Playbooks]$ vim pause.yml

[test@server3 Playbooks]$ cat pause.yml

--- # The Pause Module

- hosts: apacheweb

  sudo: yes

  gather\_facts: no

  tasks:

  - name: Install HTTPD

    action: yum name=httpd state=installed

  - name: Pausing

    pause:

      prompt: Press ENTER to Continue...

  - name: Verify lynx installation

    action: yum name=lynx state=present

*Execute the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook pause.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Install HTTPD] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [Pausing] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[server4.mylabserver.com]

Press ENTER to Continue...:

 ok: [server4.mylabserver.com]

 TASK: [Verify lynx installation] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=3    changed=0    unreachable=0    failed=0

*Run the appropriate ansible command to list all the hosts that are configured on your system for access by the control server. Display the ansible 'hosts' file in active use on the control server so that the hosts and groups are displayed.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

[test@server3 ~]$ cat /etc/ansible/hosts

[local]

localhost

 [apacheweb]

server4.mylabserver.com

[appserver]

server5.mylabserver.com

*Create a full Ansible Playbook called 'waitfor.yml' with the following characteristics, incorporating the WAITFOR module, display the playbook once created:*

*- Connect to one (or a group) of hosts from the list displayed in Step #1 above*

*- Force the playbook to run with 'sudo' privileges*

*- Skip gathering facts on the remote system(s)*

*- Using the appropriate package management module, install Apache Tomcat on the remote system(s)*

*- Using the WAITFOR Module, wait for port 8080 to start listening for connections before continuing*

*- Using the appropriate package maangement module, check to see if the 'lynx' browser is installed*

[test@server3 Playbooks]$ vim waitfor.yml

[test@server3 Playbooks]$ cat waitfor.yml

--- # The Wait For Module

- hosts: apacheweb

  sudo: yes

  gather\_facts: no

  tasks:

    - name: Installing Apache Tomcat

      action: yum name=tomcat state=installed

    - name: Waiting for Port 8080 to Listen

      wait\_for:

        port: 8080

        state: started

    - name: Verifying Lynx Installation

      action: yum name=lynx state=present

*Execute the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook waitfor.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Installing Apache Tomcat] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 TASK: [Waiting for Port 8080 to Listen] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 (NOTE WHILE WAITING, THE FOLLOWING WAS DONE ON SERVER4) -------------------

[test@server4 ~]$ ps aux | grep tomcat

test      1620  0.0  0.0 112640   964 pts/0    R+   18:01   0:00 grep --color=auto tomcat

 [test@server4 ~]$ sudo systemctl start tomcat

----------------------------------------------------------------------------------------

 TASK: [Verifying Lynx Installation] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=3    changed=1    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'setup' module for ansible, list all of the known facts on the local system.*

[test@server3 ~]$ ansible local -m setup

localhost | success >> {

    "ansible\_facts": {

        "ansible\_all\_ipv4\_addresses": [

            "172.31.107.208"

        ],

        "ansible\_all\_ipv6\_addresses": [

            "fe80::103f:68ff:fefe:2bd1"

        ],

        "ansible\_architecture": "x86\_64",

        "ansible\_bios\_date": "05/06/2015",

        "ansible\_bios\_version": "4.2.amazon",

        "ansible\_cmdline": {

            "BOOT\_IMAGE": "/boot/vmlinuz-3.10.0-229.14.1.el7.x86\_64",

        (NOTE: Your output will be larger as this was concatenated for space and readability)

*Using the 'setup' module for ansible, list all of the known facts for all systems configured in the 'hosts' file on the system.*

[test@server3 ~]$ ansible all -m setup

(NOTE: Localhost truncated here... starting with another host output)

server4.mylabserver.com | success >> {

    "ansible\_facts": {

        "ansible\_all\_ipv4\_addresses": [

            "172.31.108.45"

        ],

        "ansible\_all\_ipv6\_addresses": [

            "fe80::1029:fbff:fed6:9aaf"

        ],

        "ansible\_architecture": "x86\_64",

        "ansible\_bios\_date": "05/06/2015",

        "ansible\_bios\_version": "4.2.amazon",

        "ansible\_cmdline": {

            "BOOT\_IMAGE": "/boot/vmlinuz-3.10.0-229.14.1.el7.x86\_64",

   (NOTE: Your output will be larger as this was concatenated for space and readability)

*While listing the facts on the local system, filter the list showing only the content pertaining to the system IP addresses.*

[test@server3 ~]$ ansible local -m setup -a 'filter=ans\*ipv4\*'

localhost | success >> {

    "ansible\_facts": {

        "ansible\_all\_ipv4\_addresses": [

            "172.31.107.208"

        ],

        "ansible\_default\_ipv4": {

            "address": "172.31.107.208",

            "alias": "eth0",

            "gateway": "172.31.96.1",

            "interface": "eth0",

            "macaddress": "12:3f:68:fe:2b:d1",

            "mtu": 9001,

            "netmask": "255.255.240.0",

            "network": "172.31.96.0",

            "type": "ether"

        }

    },

    "changed": false

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a playbook, using the 'yum' module that accomplishes the following:*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- The playbook runs as 'sudo'*

*- Updates ALL packages on the remote system*

[test@server3 Playbooks]$ vim yum.yml

[test@server3 Playbooks]$ cat yum.yml

--- # Yum Module Example

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  tasks:

  - name: Equivalent of YUM UPGRADE

    action: yum name=\* state=latest

*Run the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook yum.yml

PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Equivalent of YUM UPGRADE] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=1    changed=0    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a playbook, using the 'apt' module that accomplishes the following:*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- The playbook runs as 'sudo'*

*- Installs the Apache web server*

[test@server3 Playbooks]$ vim apt.yml

[test@server3 Playbooks]$ cat apt.yml

--- # APT MODULE EXAMPLE

- hosts: aptserver

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  tasks:

  - name: Install Apache Web Server

    apt: name=apache2 state=present update\_cache=yes

*Run the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook apt.yml

 PLAY [aptserver] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK: [Install Apache Web Server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server1.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server1.mylabserver.com      : ok=1    changed=1    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a playbook, using the 'service' module that accomplishes the following:*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- The playbook runs as 'sudo'*

*- Installs the apache web server*

*- Enables the apache web server service so it begins on reboot*

*- Starts the apache web service*

[test@server3 Playbooks]$ cat service.yml

--- # SERVICE MODULE EXAMPLE

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  tasks:

  - name: Install Web Server

    action: yum name=httpd state=installed

  - name: Start the Web Server

    service: name=httpd state=started

  - name: Enable HTTPD After Reboot

    service: name=httpd enabled=yes

*Run the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook service.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [Install Web Server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 TASK: [Start the Web Server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 TASK: [Enable HTTPD After Reboot] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=4    changed=3    unreachable=0    failed=0

 [test@server3 Playbooks]$ telnet server4 80

Trying 54.175.189.40...

Connected to server4.

Escape character is '^]'.

helo

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">

<html><head>

<title>501 Not Implemented</title>

</head><body>

<h1>Not Implemented</h1>

<p>helo to / not supported.<br />

</p>

</body></html>

Connection closed by foreign host.

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a playbook, using the 'service' module that accomplishes the following:*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- The playbook runs as 'sudo'*

*- Copies a file from a local directory called 'files' and a file called 'text4.txt' (created separately) to the /home/test directory remotely*

[test@server3 Playbooks]$ vim copy.yml

[test@server3 Playbooks]$ cat copy.yml

--- # COPY MODULE EXAMPLE

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  tasks:

  - name: Copy from the files directory test file

    action: copy src=files/test4.txt dest=/home/test/test4.txt owner=test group=test mode=0655 backup=yes

[test@server3 Playbooks]$ cd files

[test@server3 files]$ echo "this is text 4" > text4.txt

*Run the playbook and display the results.*

[test@server3 Playbooks]$ ansible-playbook copy.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Copy from the files directory test file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=1    changed=1    unreachable=0    failed=0

(SECOND SERVER)

[test@server4 ~]$ ll

total 4

drwxr-xr-x. 2 test test  6 Sep 19 19:33 playbooks

-rw-r-xr-x. 1 test test 10 Oct  6 15:13 test4.txt

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a playbook, using the 'service' module that accomplishes the following:*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- The playbook runs as 'sudo'*

*- Runs a shell script called 'test.sh' in the remote directory of /home/test/testing (created separately)*

[test@server3 Playbooks]$ vim command.yml

[test@server3 Playbooks]$ cat command.yml

--- # COMMAND MODULE EXAMPLE

- hosts: appserver

  user: test

  sudo: yes

  connection: ssh

  gather\_facts: no

  tasks:

    - name: Check for python packages

      command: /home/test/testing/test.sh

      args:

        chdir: /home/test/testing

(SECOND SERVER)

[test@server5 testing]$ cat test.sh

#!/bin/bash

 echo "This is a message" > output.txt

  Run the playbook and display the results.

[test@server3 Playbooks]$ ansible-playbook command.yml

PLAY [appserver] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 TASK: [Check for python packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server5.mylabserver.com]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server5.mylabserver.com      : ok=1    changed=1    unreachable=0    failed=0

(SECOND SERVER)

[test@server5 testing]$ ll

total 8

-rw-r--r--. 1 root root 18 Oct  6 15:17 output.txt

-rwxr-xr-x. 1 test test 51 Oct  3 21:34 test.sh

***Module 5***

***Questions***

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a directory structure using the content directions from the Roles series as follows:*

*- Top level 'roles' directory*

*- Roles that include 'webservers', 'appservers', 'cachingservers', 'common'*

*- Add the appropriate subdirectories so that the automatic include of the 'main.yml' file, if it exists, is included in any master control playbook*

[test@server3 ]$ mkdir roles

[test@server3 ]$ cd roles

[test@server3 roles]$ mkdir appservers cachingservers common webservers

[test@server3 roles]$ mkdir appservers/defaults appservers/files appservers/handlers appservers/meta appservers/tasks appservers/templates appservers/vars

<NOTE REPEAT ABOVE COMMAND FOR EACH TOP LEVEL ROLE>

*List the directory structure of 'webservers' (which should be identical to the remaining).*

[test@server3 roles]$ ls -al webservers/

total 4

drwxrwxr-x. 9 test test   98 Oct 26 23:43 .

drwxrwxr-x. 8 test test 4096 Oct 27 21:26 ..

drwxrwxr-x. 2 test test    6 Oct 24 21:00 defaults

drwxrwxr-x. 2 test test    6 Oct 24 21:00 files

drwxrwxr-x. 2 test test   21 Oct 27 20:59 handlers

drwxrwxr-x. 2 test test    6 Oct 24 21:00 meta

drwxrwxr-x. 2 test test   21 Oct 27 21:00 tasks

drwxrwxr-x. 2 test test    6 Oct 24 21:00 templates

drwxrwxr-x. 2 test test   21 Oct 27 00:26 vars

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a directory structure using the content directions from the Roles series and in our previous exercises, in the subdirectory called 'tasks' do the following:*

*- Create a YAML File called 'main.yml'*

*- Add two named tasks to that file that install two packages (of any kind)*

*- Create a top level playbook called 'webservers.yml' that has the appropriate user (test), sudo rights, gathering facts set to no and runs over SSH and includes the 'webservers' role we created earlier  whose 'tasks' directory we added the main plays to in the above step.*

[test@server3 roles]$ vim webservers/tasks/main.yml

[test@server3 roles]$ cat webservers/tasks/main.yml

- name: Install (Red Hat/CentOS) Apache Web Server

  yum: pkg=httpd state=latest

- name: Install (Red Hat/CentOS) Apache Web Server

  yum: pkg=telnet state=latest

*Run the playbook and display the results.*

[test@server3 roles]$ ansible-playbook webservers.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

TASK: [When did the ROLE start] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [redhat\_webservers | Install (Red Hat/CentOS) Apache Web Server] \*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=2    changed=0    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a directory structure using the content directions from the Roles series and in our previous exercises, in the subdirectory called 'handlers' do the following:*

*- Create a YAML File called 'main.yml'*

*- Add a named task that uses the 'service' module to restart the service created from our previous exercise (httpd)*

*- Create a top level playbook called 'webservers.yml' that has the appropriate user (test), sudo rights, gathering facts set to no and runs over SSH and includes the 'webservers' role we created earlier  whose 'handlers' directory we added the main plays to in the above step.*

[test@server3 roles]$ cat webservers/handlers/main.yml

- name: Restart HTTPD

  service: name=httpd state=restarted

[test@server3 roles]$ cat webservers.yml

--- # Master Playbook for Web Servers

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  roles:

  - webservers

*Run the playbook and display the results.*

[test@server3 roles]$ ansible-playbook webservers.yml

PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [When did the ROLE start] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

TASK: [redhat\_webservers | Install (Red Hat/CentOS) Apache Web Server] \*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [redhat\_webservers | Restarting the HTTPD Service] \*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=2    changed=0    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a directory structure using the content directions from the Roles series and in our previous exercises, in the subdirectory called 'handlers' do the following:*

*- Create a YAML File called 'main.yml'*

*- Add a named task that uses the 'service' module to restart the service created from our previous exercise (httpd)*

*- Create a top level playbook called 'webservers.yml' that has the appropriate user (test), sudo rights, gathering facts set to no and runs over SSH and includes the 'webservers' role we created earlier  whose 'handlers' directory we added the main plays to in the above step.*

[test@server3 roles]$ cat webservers/handlers/main.yml

- name: Restart HTTPD

  service: name=httpd state=restarted

[test@server3 roles]$ cat webservers.yml

--- # Master Playbook for Web Servers

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  roles:

  - webservers

*Run the playbook and display the results.*

[test@server3 roles]$ ansible-playbook webservers.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [When did the ROLE start] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [redhat\_webservers | Install (Red Hat/CentOS) Apache Web Server] \*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

TASK: [redhat\_webservers | Restarting the HTTPD Service] \*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=2    changed=0    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create a master playbook that accomplishes the following:*

*- Runs against a server/group of servers defined in your hosts configuration*

*- Runs with the user 'test'*

*- Executes with 'sudo' privileges*

*- Includes the 'webservers' role from our previous examples*

*- Only executes the role if the OS is RedHat or CentOS*

[test@server3 roles]$ cat webservers.yml

--- # Master Playbook for Web Servers

- hosts: apacheweb

  user: test

  sudo: yes

  connection: ssh

  roles:

  - { role: webservers, when: "ansible\_os\_family == 'RedHat'" }

*Run the playbook and display the results*

[test@server3 roles]$ ansible-playbook webservers.yml

 PLAY [apacheweb] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [When did the ROLE start] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 TASK: [redhat\_webservers | Install (Red Hat/CentOS) Apache Web Server] \*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 TASK: [redhat\_webservers | shell systemctl status httpd] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [server4.mylabserver.com]

 TASK: [webservers | debug var=result] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com] => {

    "var": {

        "result": {

            "attempts": 1,

            "changed": true,

            "cmd": "systemctl status httpd",

            "delta": "0:00:00.007100",

            "end": "2015-11-03 16:04:42.389739",

            "invocation": {

                "module\_args": "systemctl status httpd",

                "module\_name": "shell"

            },

            "rc": 0,

            "start": "2015-11-03 16:04:42.382639",

            "stderr": "",

            "stdout": "httpd.service - The Apache HTTP Server\n   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)\n   Active: active (running) since Tue 2015-11-03 16:04:39 UTC; 2s ago\n Main PID: 2228 (httpd)\n   Status: \"Processing requests...\"\n   CGroup: /system.slice/httpd.service\n           |-2228 /usr/sbin/httpd -DFOREGROUND\n           |-2229 /usr/sbin/httpd -DFOREGROUND\n           |-2230 /usr/sbin/httpd -DFOREGROUND\n           |-2231 /usr/sbin/httpd -DFOREGROUND\n           |-2232 /usr/sbin/httpd -DFOREGROUND\n           `-2233 /usr/sbin/httpd -DFOREGROUND\n\nNov 03 16:04:39 server4.mylabserver.com systemd[1]: Starting The Apache HTTP Server...\nNov 03 16:04:39 server4.mylabserver.com systemd[1]: Started The Apache HTTP Server.",

            "stdout\_lines": [

                "httpd.service - The Apache HTTP Server",

                "   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)",

                "   Active: active (running) since Tue 2015-11-03 16:04:39 UTC; 2s ago",

                " Main PID: 2228 (httpd)",

                "   Status: \"Processing requests...\"",

                "   CGroup: /system.slice/httpd.service",

                "           |-2228 /usr/sbin/httpd -DFOREGROUND",

                "           |-2229 /usr/sbin/httpd -DFOREGROUND",

                "           |-2230 /usr/sbin/httpd -DFOREGROUND",

                "           |-2231 /usr/sbin/httpd -DFOREGROUND",

                "           |-2232 /usr/sbin/httpd -DFOREGROUND",

                "           `-2233 /usr/sbin/httpd -DFOREGROUND",

                "",

                "Nov 03 16:04:39 server4.mylabserver.com systemd[1]: Starting The Apache HTTP Server...",

                "Nov 03 16:04:39 server4.mylabserver.com systemd[1]: Started The Apache HTTP Server."

            ],

            "warnings": []

        }

    }

}

 TASK: [webservers | debug Need to install telnet] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com] => {

    "msg": "Hello world!"

}

TASK: [debian\_webservers | Install (Red Hat/CentOS) Apache Web Server] \*\*\*\*\*\*\*\*

skipping: [server4.mylabserver.com]

 TASK: [When did the ROLE end] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server4.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server4.mylabserver.com      : ok=5    changed=2    unreachable=0    failed=0

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Copy the template created during our previous exercises to the local role directory for the role you are using (put it into the appropriate directory within the role).*

[test@server3 roles]$ ls -al debian\_webservers/templates/

total 4

drwxrwxr-x. 2 test test  25 Nov 11 03:12 .

drwxrwxr-x. 9 test test  98 Oct 27 21:22 ..

-rw-rw-r--. 1 test test 297 Nov 11 03:06 test.conf.j2

*Create a master playbook that does the following to verify the template is used as expected:*

*- Runs against one or more of the servers/group in Step #2 above*

*- Runs as the ansible user*

*- Executes with sudo privileges*

*- Uses SSH*

*- Runs the role from step #3*

[test@server3 roles]$ cat testweb.yml

--- # Master Playbook for Web Servers

- hosts: debian

  user: test

  sudo: yes

  connection: ssh

  roles:

    - debian\_webservers

*Execute the playbook, making sure to pass in whatever variables are needed from the template created*

[test@server3 roles]$ ansible-playbook testweb.yml --extra-vars "connectionType=FTP userName=test userPassword=password123"

 PLAY [debian] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [server1.mylabserver.com]

 TASK: [debian\_webservers | Install the template configuration file customized for the remote distribution] \*\*\*

changed: [server1.mylabserver.com]

 PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

server1.mylabserver.com      : ok=2    changed=1    unreachable=0    failed=0

***Module 6***

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'ansible' command line utility, execute a ONE LINE ansible command that does the following:*

*- Runs against the server/group chosen in Step #2*

*- Uses the 'test' user to run the command*

*- Executes the command with 'sudo' privileges*

*- Uses one of the modules for executing shell commands*

*- List all the files in the remote system(s) /var directory*

[test@server3 Outline]$ ansible apacheweb -u test -s -m command -a "ls -al /var"

*Display the results.*

server4.mylabserver.com | success | rc=0 >>  
total 16  
drwxr-xr-x. 20 root root 4096 Nov  2 23:57 .  
drwxr-xr-x. 17 root root 4096 Aug  6 19:28 ..  
drwxr-xr-x.  2 root root    6 Jun 10  2014 adm  
drwxr-xr-x.  9 root root   91 Nov  2 23:57 cache  
drwxr-xr-x.  2 root root    6 May 12 20:18 crash  
drwxr-xr-x.  3 root root   32 Aug  6 19:40 db  
drwxr-xr-x.  3 root root   17 Sep 29  2014 empty  
drwxr-xr-x.  2 root root    6 Jun 10  2014 games  
drwxr-xr-x.  2 root root    6 Jun 10  2014 gopher  
drwxr-xr-x.  3 root root   17 Sep 15 13:33 kerberos  
drwxr-xr-x. 42 root root 4096 Nov  2 23:57 lib  
drwxr-xr-x.  2 root root    6 Jun 10  2014 local  
lrwxrwxrwx.  1 root root   11 Sep 29  2014 lock -> ../run/lock  
drwxr-xr-x. 12 root root 4096 Nov  2 23:20 log  
lrwxrwxrwx.  1 root root   10 Sep 29  2014 mail -> spool/mail  
drwxr-xr-x.  2 root root    6 Jun 10  2014 nis  
drwxr-xr-x.  2 root root    6 Jun 10  2014 opt  
drwxr-xr-x.  2 root root    6 Jun 10  2014 preserve  
lrwxrwxrwx.  1 root root    6 Sep 29  2014 run -> ../run  
drwxr-xr-x. 10 root root  102 Oct  6 22:47 spool  
drwxrwxrwt.  4 root root   64 Nov  2 23:57 tmp  
drwxr-xr-x.  3 root root   16 Sep 29  2014 var  
drwxr-xr-x.  2 root root    6 Jun 10  2014 yp

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'ansible' command line utility, execute a ONE LINE ansible command that does the following:*

*- Runs against the server/group chosen in Step #2*

*- Uses the 'test' user to run the command*

*- Executes the command with 'sudo' privileges*

*- Uses one of the modules for installing packages*

*- Install the package 'lynx' on the remote server(s)*

[test@server3 roles]$ ansible apacheweb -u test -s -m yum -a "pkg=lynx state=latest"

*Display the results.*

server4.mylabserver.com | success >> {

    "changed": true,

    "msg": "",

    "rc": 0,

    "results": [

        "Loaded plugins: fastestmirror\nLoading mirror speeds from cached hostfile\n \* base: mirror.symnds.com\n \* extras: mirror.cogentco.com\n \* updates: mirrors.advancedhosters.com\nResolving Dependencies\n--> Running transaction check\n---> Package lynx.x86\_64 0:2.8.8-0.3.dev15.el7 will be installed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n================================================================================\n Package       Arch            Version                      Repository     Size\n================================================================================\nInstalling:\n lynx          x86\_64          2.8.8-0.3.dev15.el7          base          1.4 M\n\nTransaction Summary\n================================================================================\nInstall  1 Package\n\nTotal download size: 1.4 M\nInstalled size: 5.4 M\nDownloading packages:\nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\n  Installing : lynx-2.8.8-0.3.dev15.el7.x86\_64                              1/1 \n  Verifying  : lynx-2.8.8-0.3.dev15.el7.x86\_64                              1/1 \n\nInstalled:\n  lynx.x86\_64 0:2.8.8-0.3.dev15.el7                                             \n\nComplete!\n"

    ]

}

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'ansible' command line utility, execute a ONE LINE ansible command that does the following:*

*- Runs against the server/group chosen in Step #2*

*- Uses the 'test' user to run the command*

*- Executes the command with 'sudo' privileges*

*- Uses the 'service' module, start the HTTPD service installed during an earlier exercise*

[test@server3 roles]$ ansible apacheweb -u test -s -m service -a "name=httpd state=started"

*Display the results.*

server4.mylabserver.com | success >> {

    "changed": true,

    "name": "httpd",

    "state": "started"

}

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'ansible' command line utility, execute a ONE LINE ansible command that does the following:*

*- Runs against the server/group chosen in Step #2*

*- Uses the 'test' user to run the command*

*- Executes the command with 'sudo' privileges*

*- Uses the 'user' module to add a user called 'johnsmith' with the default shell being 'bash' and indicate a unique UID*

[test@server3 ~]$ ansible redhat -u test -s -m user -a "name=johnsmith uid=25001 shell=/bin/bash"

*Display the results.*

server4.mylabserver.com | success >> {

    "changed": true,

    "comment": "",

    "createhome": true,

    "group": 25001,

    "home": "/home/johnsmith",

    "name": "johnsmith",

    "shell": "/bin/bash",

    "state": "present",

    "system": false,

    "uid": 25001

}

localhost | success >> {

    "changed": true,

    "comment": "",

    "createhome": true,

    "group": 25001,

    "home": "/home/johnsmith",

    "name": "johnsmith",

    "shell": "/bin/bash",

    "state": "present",

    "system": false,

    "uid": 25001

}

server5.mylabserver.com | success >> {

    "changed": true,

    "comment": "",

    "createhome": true,

    "group": 25001,

    "home": "/home/johnsmith",

    "name": "johnsmith",

    "shell": "/bin/bash",

    "state": "present",

    "system": false,

    "uid": 25001

}

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'ansible' command line utility, execute a ONE LINE ansible command that does the following:*

*- Runs against the server/group chosen in Step #2*

*- Uses the 'test' user to run the command*

*- Executes the command with 'sudo' privileges*

*- Adds a root cron job with a name (you assign it) that runs a list of the /var directory on the system and logs to the root home directory in a file called 'var.log', every day of week/month at the same time*

[test@server3 ~]$ ansible redhat -u test -s -m cron -a "name='crontest' minute='0' hour='12' job='ls -al /var > /root/var.log'"

*Display the results and the resulting CRON entry*

server4.mylabserver.com | success >> {

    "changed": true,

    "jobs": [

        "crontest"

    ]

}

localhost | success >> {

    "changed": true,

    "jobs": [

        "crontest"

    ]

}

 server5.mylabserver.com | success >> {

    "changed": true,

    "jobs": [

        "crontest"

    ]

}

[test@server3 ~]$ sudo crontab -l

#Ansible: crontest

0 12 \* \* \* ls -al /var > /root/var.log

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Using the 'ansible' command line utility, execute a ONE LINE ansible command that does the following:*

*- Runs against the server/group chosen in Step #2*

*- Uses the 'test' user to run the command*

*- Executes the command with 'sudo' privileges*

*- Displays the contents of each server(s) remote /etc/fstab file locally using ONLY the command line (no module)*

[test@server3 ~]$ ansible all -u test -a "cat /etc/fstab"

*Display the results.*

localhost | success | rc=0 >>

 #

# /etc/fstab

# Created by anaconda on Mon Sep 29 21:48:54 2014

#

# Accessible filesystems, by reference, are maintained under '/dev/disk'

# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info

#

UUID=0f790447-ebef-4ca0-b229-d0aa1985d57f /                       xfs     defaults        1 1

/root/swap swap swap sw 0 0

server4.mylabserver.com | success | rc=0 >>

 #

# /etc/fstab

# Created by anaconda on Mon Sep 29 21:48:54 2014

#

# Accessible filesystems, by reference, are maintained under '/dev/disk'

# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info

#

UUID=0f790447-ebef-4ca0-b229-d0aa1985d57f /                       xfs     defaults        1 1

/root/swap swap swap sw 0 0

server5.mylabserver.com | success | rc=0 >>

 #

# /etc/fstab

# Created by anaconda on Mon Sep 29 21:48:54 2014

#

# Accessible filesystems, by reference, are maintained under '/dev/disk'

# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info

#

UUID=0f790447-ebef-4ca0-b229-d0aa1985d57f /                       xfs     defaults        1 1

/root/swap swap swap sw 0 0

/dev/xvdf1 /mnt/data ext3 rw 0 0

 server1.mylabserver.com | success | rc=0 >>

LABEL=cloudimg-rootfs   /        ext4   defaults,discard        0 0

***Module 6***

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create an outline that lists the installation and configuration process for an Apache web server(s) with the following kinds of information (add to this list as needed):*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- Creates a list of 'things we need to know' (variables)*

*- List of 'things that need to be done' (tasks)*

*- How do we test (tasks, handlers and/or debug statements)*

[test@server3 usecases]$ vim mywebserver.txt

[test@server3 usecases]$ cat mywebserver.txt

- Build an Apache Web Server

- Deploy the webserver with the ansible user

- Ansible user needs sudo

- Connection type ssh rather than paramiko

- Leave gathering of facts on

- Info Needed:

  - apache web server package name

  - distribution running on

  - apache version

  - need the web server directory customized

  - what is the site name

- What needs to be done

  - install the apache web server on the remote system(s)

  - create the web server directory

  - web server directory has proper ownership

  - configuration of apache

    - copy the http configuration file

    - set up the vhost directory

    - add any ssl keys if needed

    - add the vhost template for the default site

  - copy the site code to the new directory

  - start the apache service

  - add/enable any modules needed

    - mysql/mariadb

    - ssl

    - mod/rewrite

  - restart the apache service (or move the start to here)

 - Test

  - using lynx or telnet, test that the web service is running

    - waitfor port 80 to be listening

  - register output for the service status as JSON on the completion of the playbook

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create an outline that lists the installation and configuration process for a MySQL/MariaDB server with the following kinds of information (add to this list as needed):*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- Creates a list of 'things we need to know' (variables)*

*- List of 'things that need to be done' (tasks)*

*- How do we test (tasks, handlers and/or debug statements)*

[test@server3 usecases]$ vim mydbserver.txt

[test@server3 usecases]$ cat mydbserver.txt

- Installing and Configuring a MariaDB Server (master)

- The installation will be done with the ansible user

- the installation needs to be done with sudo privileges

- the connection used is ssh

- gathering of facts needs to be on

 - What do we need to know?

  - the package name of the DB server

  - the group/host of the master server

  - the directory for installing the db (if not default)

  - the version of the db

  - the distribution it is installed on

 - What needs to be done/installed

  - install the MariaDB server and utilities

  - root password install

    - waitfor the db service to be started

      - takes place manually after the mysql-secure-installation run

  - install the mysql/mariadb configuration file (if needed)

  - copy the mysql/mariadb database backup to the home directory

  - import the database(s)

  - add a cron job for nightly backups

 - Testing the db

  - run a MYSQL command and register the output as JSON format to determine success

*Verify that your ansible installation is available by displaying the version of ansible while logged in as the 'user' user.*

[test@server3 ~]$ ansible --version

ansible 1.9.2

  configured module search path = None

*Run the ansible command that lists all of the hosts configured in your control server 'hosts' file for the system.*

[test@server3 ~]$ ansible all --list-hosts

    server5.mylabserver.com

    localhost

    server4.mylabserver.com

*Create an outline that lists the installation and configuration process for a NFS server with the following kinds of information (add to this list as needed):*

*- Uses SSH*

*- Logs in to the remote system as 'test' user*

*- Connects to one server or group from Step #2 above*

*- Creates a list of 'things we need to know' (variables)*

*- List of 'things that need to be done' (tasks)*

*- How do we test (tasks, handlers and/or debug statements)*

[test@server3 usecases]$ vim mynfsserver.txt

[test@server3 usecases]$ cat mynfsserver.txt

- installing and configuring an NFS static content server for web use

- installation and configuration done with the ansible user

- the ansible user needs to be sudo

- gathering facts on

- connection based on ssh

 - what do we need to know?

  - distribution of the system deploying on

  - the NFS server and client package names

  - path to the shared file space

  - the server/group we are installing NFS on

- what do we need to do / install?

  - install the NFS server/client and utilities

  - export the shared directory (/etc/exports)

  - add our LA lab network (internal)

  - configure the filesystem export for read/write on known networks

    - disable any unknown network or user connectivity

  - start the NFS service

  - cron job to backup the filesystem

  - NFS client installs? - common role

    - NFS Client /etc/fstab configuration to mount the share on boot...?

 - test

  - capture the NFS server service status as a JSON output and register the result