Ansible Control Node, Inventory & Managed Nodes

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

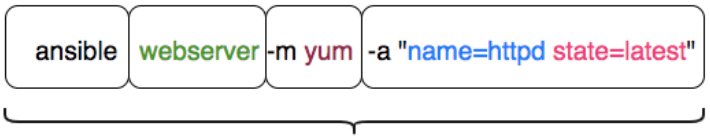
Session - 4 Agenda:

1. Ansible Ad-Hoc Commands
2. Ansible Modules

There are two ways of doing any automation using Ansible:

1. Ad-Hoc Commands: (Used for one-time tasks)
2. Ansible Playbooks: (Preferred for repetitive tasks)

AD HOC command



Ansible Playbook

```
---
- name: playbook name
  hosts: webserver
  tasks:
    - name: name of the task
      yum:
        name: httpd
        state: latest
```

We must have good knowledge and command over ad-hoc command before start learning playbooks.

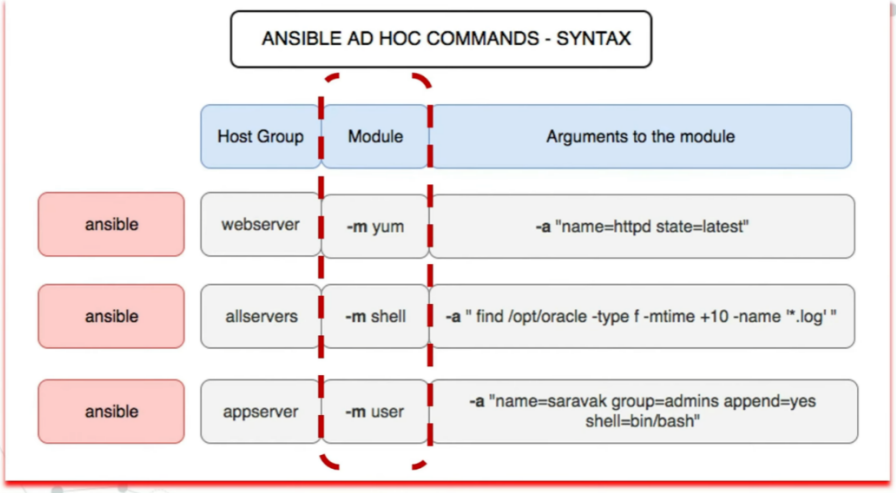
1. Ansible Ad-Hoc Commands:

An Ansible ad hoc command uses the /usr/bin/ansible command-line tool to automate a single task on one or more managed nodes. Ad-hoc commands are quick and easy, but they are not reusable.

2. Ansible Modules:

Ansible modules are units of code that can control system resources or execute system commands. Ansible provides a module library that you can execute directly on remote hosts or through playbooks. You can also write custom modules.

➢ ***Ansible Modules:***

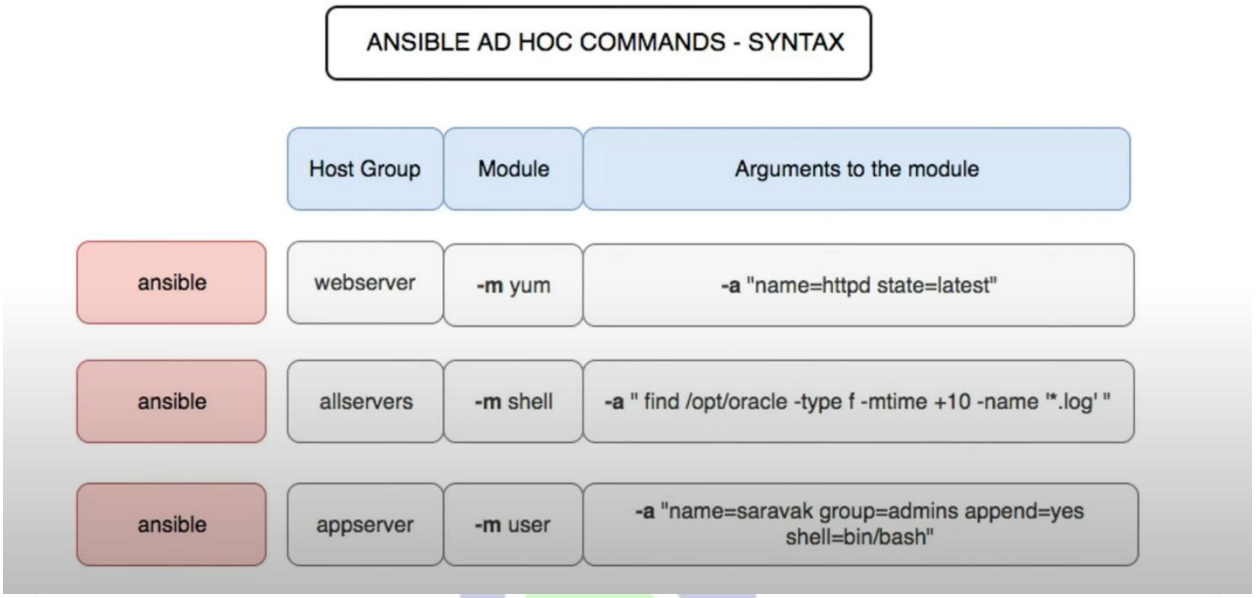


Ansible Control Node, Inventory & Managed Nodes

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Ansible Ad-Hoc Command Syntax:

Ansible ad-hoc command syntax is shown in below picture which contains host-group name, module name and argument in it.



$ ansible groupname -m modulename -a 'argument'
$ ansible groupname -m modulename -a 'key=value'

Installation of Ansible Control Node (Ansible Server):

Follow the commands in the same order to setup Ansible in your environment:

 # hostnamectl set-hostname ansible-server.nehraclasses.local
 # vim /etc/hosts
192.168.1.14 ansible-server.nehraclasses.local localhost
192.168.1.16 node1.nehraclasses.local node1
192.168.1.17 node2.nehraclasses.local node2
 # ping node1
 # ping node2
 # dnf install -y ansible-core
 # ansible --version

Create a local user account on all the nodes and add this user to sudoers file as well.

 # useradd nehraclasses
 # echo "redhat" | passwd --stdin nehraclasses
 # echo "nehraclasses ALL=(ALL) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/nehraclasses
 # su - nehraclasses

Generate SSH public and private keys on the ansible server and copy the public key file to other nodes.

$ ssh-keygen
$ ssh-copy-id nehraclasses@node1
$ ssh nehraclasses@node1
$ exit
$ ssh-copy-id nehraclasses@node2

Ansible Control Node, Inventory & Managed Nodes

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

$ ssh nehraclasses@node2
$ exit

Now, create your own custom inventory file or make an entry in default inventory file.
$ sudo vim /etc/ansible/hosts
[dev]
192.168.1.16
192.168.1.17

OR

$ mkdir ~/automation && cd ~/automation/
$ vim ansible.cfg
[defaults]
inventory = ./inventory
host_key_checking = false
remote_user = nehraclasses
ask_pass = False

[privilege_escalation]
#become=true
#become_method=sudo
#become_user=root
#become_ask_pass=False

$ vim ~/automation/inventory
[dev]
node1
node2

Verify the hosts.
$ ansible all --list-hosts

To print all the default module names with their description available in your machine.
$ ansible-doc -l

To count the number of available modules we can execute below command:
$ ansible-doc -l | wc -l

Important Ansible Module Names:

1. *Copy*
2. *Command*
3. *Raw*
4. *Shell*
5. *File*
6. *Fetch*

7. *Get_url*
8. *Lineinfile*
9. *Replace*
10. *User*
11. *Group*
12. *Yum*
13. *Package*
14. *Stat*
15. *Mount*
16. *Setup*

Using Ansible Modules:

1. Copy Module: (Copy module is used to copy files from ansible server to managed nodes.)

Example 1: Copy a file named 'nehraclasses.txt' from the PWD of ansible control node to the maneged nodes in /tmp directory.

$ echo Nehra Classes Are Awesome > nehraclasses.txt

$ ansible all -m copy -a 'src=nehraclasses.txt dest=/tmp'

Verify the file on node machines.

$ cat /tmp/nehraclasses.txt

Note: Keys (src, dest etc. in above case) in argument is already defined by the developer.

Before using any module ideally, we must go through the module documentation provided by ansible. And you can execute ansible-doc command for the module.

$ ansible-doc copy

Here we can see all the details of any module along with the examples with best use cases in the industry.

Example 2: Create file with content & copy it to managed nodes in one go.

$ ansible all -m copy -a 'content='Nehra Classes Are Awesome.' dest=/tmp/nehraclasses2.txt'

Verify the file on the node machines.

$ cat /tmp/nehraclasses2.txt

If you re-execute the same command again.

$ ansible all -m copy -a 'content="Nehra Classes Are Awesome." dest=/tmp/nehraclasses2.txt'

It will show the status in green or yellow color depending whether changes have been implemented or not.

Yellow Color => Success & changes implemented

Green Color => Success but changes not implemented

Red Color => Error

Example 3: Copy file to other location where normal users don't have write permissions.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/opt'

It will show you an error, because of not having write permission for this normal user on the destination directory.
There are many ways to deal with this problem:
(a) Using Root user account credentials.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/opt' -u root -k

And it will work for next 60 seconds even if you don't mention -u option for root user. Because by default ansible tunnel keeps itself open for 60 seconds after performing any automation (control persist).
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/opt'

Now try to create file with another name there.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/opt/nehra.txt'

(b) Using sudo privileged:
If the normal user has sudo access on all the nodes then we can use sudo access for this purpose.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/opt/file.txt' -b --become-method sudo

Or we can use this method without --become-method flag because by default it uses sudo method with become option.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/opt/file.txt' -b

To perform any job using other user account on the managed nodes.
(This is useful in many cases e.g., suppose we have some application which needs to be executed from a specific user account only which if different from that which ansible is using.)
$ ansible dev -m command -a 'id'
$ ansible dev -m command -a 'id' -b
$ ansible dev -m command -a 'id' -b --become-user vikasnehra

For repetitive root or sudo access jobs, if we don't want to use option -b in the command then we can make an entry in ansible.cfg file for the same as shown in previous session.
$ sudo vim /etc/ansible/ansible.cfg
[privilege_escalation]
become=True
#become_method=sudo
#become_user=root
#become_ask_password=False

It will always use sudo privilege after making above changes in the ansible.cfg file.
$ ansible dev -m command -a 'id'

Example 4: Copy the files to managed nodes having specified permissions, ownership & group ownership.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/root mode=0755 owner=vikasnehra group=wheel'

Verify the file on the managed nodes without going on the nodes itself.
$ ansible all -m command -a 'ls -lh /root/nehraclasses.txt'

$ ansible all -m command -a 'cat /root/nehraclasses.txt'

$ vim nehraclasses.txt
Welcome to Nehra Classes.

There is no need to mention the permissions of the same existing file, because ansible has idempotency.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/root'

Verify the changes.
$ ansible all -m command -a 'ls -lh /root/nehraclasses.txt'
$ ansible all -m command -a 'cat /root/nehraclasses.txt'

Example 5: Copy a file from one location in the managed node to other location of the managed node like using 'cp command' in linux.
$ ansible dev -m copy -a 'src=/etc/redhat-release dest=/tmp remote_src=yes'

Verify the same.
$ ansible all -m command -a 'ls -lh /tmp/redhat-release'
$ ansible all -m command -a 'cat /tmp/redhat-release'

Example 6: Taking backup before overwriting the file while copying from ansible control node to managed nodes.
$ ansible all -m copy -a 'src=nehraclasses.txt dest=/root backup=yes'

Verify the same from managed nodes.
$ ansible all -m command -a 'ls -lh /root'

---

2. Command Module: (executes a remote command on target host, in the same shell of other playbook's tasks.) Useful in case if you don't know the module name for any automation.
$ ansible dev -m command -a 'uptime'
$ ansible dev -m command -a 'lsblk'

---

3. Raw Module: (executes low-level commands where interpreter is missing on target host, a common use case is for installing python.)
$ ansible dev -m raw -a 'uptime'
$ ansible dev -m raw -a 'uptime'

Command Module Vs Raw Module:
(a) Command module will work only if python having version 2.4 or later is already installed on the managed nodes.

(b) Command module can't execute multiple commands in one go.
$ ansible dev -m command -a 'uptime ; lsblk'

But raw module can execute multiple modules.
$ ansible dev -m raw -a 'uptime ; lsblk'

---

4. Shell Module: (Executes a remote command on target host, opening a new shell /bin/sh)

Create a script file.
$ vim test.sh
#!/bin/bash
echo "Hello World"

Copy the script to all the nodes machines.
$ ansible all -m copy -a 'src=test.sh dest=/root mode=755'

Execute them from control node itself using shell module.
$ ansible all -m shell -a '/root/test.sh'

{rc=0} means return code zero.
If the return code is zero then the command executed and we got some output.

---

5. File Module: (Used for files and directories. Which we do with commands like touch, mkdir, chmod, chown, chgrp, ln, rm)
Create a directory with name as data in /tmp using file module.
$ ansible dev -m file -a 'path=/tmp/data state=directory'

If there is some mistake in argument it will show you the options as well.
$ ansible dev -m file -a 'path=/tmp/data state=dirsdhbhc'
$ ansible dev -m file -a 'path=/tmp/data state=absent'
$ ansible dev -m file -a 'path=/tmp/data state=directory mode=0777 owner=root group=root'

For more details we can execute below command to open the file module document.
$ ansible-doc file

---

6. Fetch Module: (Opposite of copy module) Used to copy files from managed nodes to ansible control node.
$ ansible all -m fetch -a 'src=/etc/redhat-release dest=backup'

Verify the same in backup folder on the control node.
$ ls -lh backup ; cd backup ; ll ; cd node1 ; ll ; cd etc ; ll

---

7. Get_url Module: (Ansible get_url module is one amongst them which helps us to access the remote URL (HTTP/HTTPS/FTP) and download a file/package from there and save it locally.)
$ ansible all -m get_url -a 'url=https://linux-training.be/linuxfun.pdf dest=/tmp/linuxfun.pdf'
$ ansible all -m get_url -a 'url=https://t.me/NehraClasses/1574 dest=/tmp/grub.txt'
$ ansible-doc get_url

Thank You