<https://www.server-world.info/en/note?os=CentOS_6&p=ansible&f=1>

$ sudo apt-get update

$ sudo apt-get install software-properties-common

$ sudo apt-add-repository ppa:ansible/ansible

$ sudo apt-get update

$ sudo apt-get install ansible

**Playbook1:**

**Requirement:**

1. Create file at /tmp as vpt

---

- hosts: all

tasks:

- name: create file in /tmp

file: path=/tmp/vpt\_devops state=touch

...

to check playbook syntax use below command

**ansible-playbook playbookname.yml --syntax-check**

**Playbook2:**

**Requiremnt:**

1. Create file at /opt/vpt\_aws only on redhat family
2. Create directory at /tmp/vpt\_devops only on debian family

---

- hosts: all

tasks:

- name: create file on redhat family

file: path=/opt/vpt\_aws state=touch

when: ansible\_os\_family == "RedHat"

- name: create dir on debian family

file: path=/tmp/opt\_devops state=directory

when: ansible\_os\_family == "Debian"

**...**

**Ececute playbook using below command**

# ansible-playbook file\_dir.yml

**Output:**

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

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server]

ok: [client1]

TASK [create file on redhat family] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

changed: [client1]

TASK [create dir on debian family] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [client1]

changed: [ansi-server]

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

ansi-server : ok=2 changed=1 unreachable=0 failed=0

client1 : ok=2 changed=1 unreachable=0 failed=0

**Playbook3:**

**Requirement:**

1. Install Webserver like apache2 or httpd on all the hosts and start the services.

# vi webserver.yml

---

- hosts: all

tasks:

- name: install webserver on redhat family

yum: pkg=httpd state=present

when: ansible\_os\_family == "Redhat"

- name: start httpd services on redhat family

service: name=httpd state=started

when: ansible\_os\_family == "Redhat"

- name: install webserver on debian family

apt: pkg=apache2 state=present

when: ansible\_os\_family == "Debian"

- name: start apache2 services on debian family

service: name=apache2 state=started

when: ansible\_os\_family == "Debian"

...

# ansible-playbook webserver.yml

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

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server]

ok: [client1]

TASK [install webserver on redhat family] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

skipping: [client1]

TASK [start httpd services on redhat family] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

skipping: [client1]

TASK [install webserver on debian family] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [client1]

ok: [ansi-server]

TASK [start apache2 services on debian family] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [client1]

ok: [ansi-server]

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

ansi-server : ok=3 changed=0 unreachable=0 failed=0

client1 : ok=1 changed=0 unreachable=0 failed=0

Playbook:4

Requirement: install nginx and apache2 on different os families using ‘and’ , ‘or’ operators.

# vi webserver.yml

---

- hosts: all

tasks:

- name: find hostname

raw: hostname

register: host

- debug: msg="{{host.stdout}}"

- name: find home path for nginx

stat:

path=/usr/share

register: path

- name: install nginx on redhat

yum: pkg=nginx state=present

when: path.stat.exists == true and host.stdout == "ansi-client\r\n" or host.stdout == "ansi-client\n"

- name: start nginx service

service: name=nginx state=started

when: path.stat.exists == true and host.stdout == "ansi-client\r\n" or host.stdout == "ansi-client\n"

- name: start nginx service

service: name=nginx state=started

when: path.stat.exists == true and host.stdout == "ansi-client\r\n" or host.stdout == "ansi-client\n"

- name: find home path for apache2

stat:

path=/var

register: ap2

- name: install apache2 on debian family

apt: pkg=apache2 state=present

when: ap2.stat.exists == true and host.stdout == "ansi-server\r\n" or host.stdout == "ansi-server\n"

- name: start apache2 service

service: name=apache2 state=started

when: ap2.stat.exists == true and host.stdout == "ansi-server\r\n" or host.stdout == "ansi-server\n"

# ansible-playbook webserver.yml

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

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server]

ok: [ansi-client]

TASK [find hostname] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [ansi-server]

changed: [ansi-client]

TASK [debug] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server] => {

"msg": "ansi-server\n"

}

ok: [ansi-client] => {

"msg": "ansi-client\r\n"

}

TASK [find home path for nginx] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server]

ok: [ansi-client]

TASK [install nginx on redhat] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

ok: [ansi-client]

TASK [start nginx service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

ok: [ansi-client]

TASK [find home path for apache2] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server]

ok: [ansi-client]

TASK [install apache2 on debian family] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-client]

ok: [ansi-server]

TASK [start apache2 service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-client]

ok: [ansi-server]

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

ansi-client : ok=7 changed=1 unreachable=0 failed=0

ansi-server : ok=7 changed=1 unreachable=0 failed=0

**Looping statements:**

Often you’ll want to do many things in one task, such as create a lot of users, install a lot of packages, or repeat a polling step until a certain result is reached.

**Playbook:**

**Reguirements:**  Install multiple packages on redhat and debian family , creation of multiple users with password, creation of multiple directories at a time.

# vi loops.yml

---

- hosts: all

tasks:

- name: perform multiple tasks using looping statements on redhat family

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

with\_items:

- httpd

- nginx

- php

- mysql

- tree

when: ansible\_os\_family == "RedHat"

- name: install multiple packages using looping statements on debian family

apt: pkg={{item}} state=present

with\_items:

- apache2

- git

when: ansible\_os\_family == "Debian"

- name: create multiple users at single time

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

with\_items:

- devops-vpt

- aws-ansible

- devops-jenkins

- devops-docker

- name: creation of multiple directories in /tmp using loops

file: path=/tmp/{{item}} state=directory mode=0777 owner=docker

with\_items:

- e

- f

- g

- h

**Output:**

root@ansi-server:/etc/ansible/playbooks# ansible-playbook loops.yml

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

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server]

ok: [ansi-client]

TASK [perform multiple tasks using looping statements on redhat family] \*\*\*\*\*\*\*\*

skipping: [ansi-server] => (item=[])

changed: [ansi-client] => (item=[u'httpd', u'nginx', u'php', u'mysql', u'tree'])

TASK [install multiple packages using looping statements on debian family] \*\*\*\*\*

skipping: [ansi-client] => (item=[])

changed: [ansi-server] => (item=[u'apache2', u'git'])

TASK [create multiple users at single time] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [ansi-server] => (item=devops-vpt)

changed: [ansi-client] => (item=devops-vpt)

changed: [ansi-server] => (item=aws-ansible)

changed: [ansi-client] => (item=aws-ansible)

changed: [ansi-server] => (item=devops-jenkins)

changed: [ansi-server] => (item=devops-docker)

changed: [ansi-client] => (item=devops-jenkins)

changed: [ansi-client] => (item=devops-docker)

TASK [creation of multiple directories in /tmp using loops] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [ansi-server] => (item=e)

changed: [ansi-client] => (item=e)

changed: [ansi-server] => (item=f)

changed: [ansi-server] => (item=g)

changed: [ansi-client] => (item=f)

changed: [ansi-server] => (item=h)

changed: [ansi-client] => (item=g)

changed: [ansi-client] => (item=h)

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

ansi-client : ok=4 changed=3 unreachable=0 failed=0

ansi-server : ok=4 changed=3 unreachable=0 failed=0

**Ansible adhoc commands:**

What’s an ad-hoc command?

An ad-hoc command is something that you might type in to do something really quick, but don’t want to save for later.

This is a good place to start to understand the basics of what Ansible can do prior to learning the playbooks language – ad-hoc commands can also be used to do quick things that you might not necessarily want to write a full playbook for.

Generally speaking, the true power of Ansible lies in playbooks. Why would you use ad-hoc tasks versus playbooks?

For instance, if you wanted to power off all of your lab for Christmas vacation, you could execute a quick one-liner in Ansible without writing a playbook.

For configuration management and deployments, though, you’ll want to pick up on using ‘/usr/bin/ansible-playbook’ – the concepts you will learn here will port over directly to the playbook language.

**Getting started:**

**Requirements:**

1. Copy files from source to destination

# ansible all -m copy -a "src=/etc/ansible/playbooks/host.yml dest=/tmp"

Above command ‘all’ represents hosts information, ‘-m’ represents module, ‘-a’ arguments, src= source file location, dest= destionation location.

1. Set file permissions using adhoc commands,

# ansible all -m file -a "dest=/tmp/host.yml mode=777"

Here destonatriuon location file should be exist.

1. Creation of directories using adhoc commands

# ansible all -m file -a "dest=/mnt/devops mode=766 owner=root group=root state=directory"

1. Deletion of duirectories using below command

# ansible all -m file -a "dest=/mnt/devops state=absent"

1. Install package using commands

# ansible dev -m yum -a "name=httpd state=present"

**Output:**

ansi-client | SUCCESS => {

"changed": false,

"msg": "",

"rc": 0,

"results": [

"httpd-2.2.34-1.16.amzn1.x86\_64 providing httpd is already installed"

]

}

1. Uninstall package using commands

# ansible dev -m yum -a "name=httpd state=absent"

Output:

ansi-client | SUCCESS => {

"changed": true,

"msg": "",

"rc": 0,

"results": [

"Loaded plugins: priorities, update-motd, upgrade-helper\nResolving Dependencies\n--> Running transaction check\n---> Package httpd.x86\_64 0:2.2.34-1.16.amzn1 will be erased\n--> Processing Dependency: httpd-mmn = 20051115 for package: php-5.3.29-1.8.amzn1.x86\_64\n--> Running transaction check\n---> Package php.x86\_64 0:5.3.29-1.8.amzn1 will be erased\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n================================================================================\n Package Arch Version Repository Size\n================================================================================\nRemoving:\n httpd x86\_64 2.2.34-1.16.amzn1 @amzn-updates 3.1 M\nRemoving for dependencies:\n php x86\_64 5.3.29-1.8.amzn1 @amzn-main 7.4 M\n\nTransaction Summary\n================================================================================\nRemove 1 Package (+1 Dependent package)\n\nInstalled size: 10 M\nDownloading packages:\nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\n Erasing : php-5.3.29-1.8.amzn1.x86\_64 1/2 \n Erasing : httpd-2.2.34-1.16.amzn1.x86\_64 2/2 \n Verifying : httpd-2.2.34-1.16.amzn1.x86\_64 1/2 \n Verifying : php-5.3.29-1.8.amzn1.x86\_64 2/2 \n\nRemoved:\n httpd.x86\_64 0:2.2.34-1.16.amzn1 \n\nDependency Removed:\n php.x86\_64 0:5.3.29-1.8.amzn1 \n\nComplete!\n"

]

}

1. Creation of users using commands

# ansible all -m user -a "name=dev password=abc123"

Output:

ansi-server | SUCCESS => {

"changed": true,

"comment": "",

"create\_home": true,

"group": 1009,

"home": "/home/dev",

"name": "dev",

"password": "NOT\_LOGGING\_PASSWORD",

"shell": "",

"state": "present",

"system": false,

"uid": 1009

}

ansi-client | SUCCESS => {

"changed": true,

"comment": "",

"create\_home": true,

"group": 509,

"home": "/home/dev",

"name": "dev",

"password": "NOT\_LOGGING\_PASSWORD",

"shell": "/bin/bash",

"state": "present",

"system": false,

"uid": 509

}

1. Deletion of users using commands

# ansible all -m user -a "name=dev state=absent"

ansi-server | SUCCESS => {

"changed": true,

"force": false,

"name": "dev",

"remove": false,

"state": "absent"

}

ansi-client | SUCCESS => {

"changed": true,

"force": false,

"name": "dev",

"remove": false,

"state": "absent"

}

1. Starting services using commands

# ansible all -m service -a "name=nginx state=started"

Output:

ansi-client | SUCCESS => {

"changed": true,

"name": "nginx",

"state": "started"

}

1. Stop the service using commands

# ansible all -m service -a "name=nginx state=stopped"

Output:

ansi-client | SUCCESS => {

"changed": true,

"name": "nginx",

"state": "stopped"

}

**Ansible-Vault:**

Ansible Vault is a feature of ansible that allows you to keep sensitive data such as passwords or keys in encrypted files, rather than as plaintext in playbooks or roles. These vault files can then be distributed or placed in source control.

1. Encrypt existing playbooks

# ansible-vault encrypt service.yml

Output:

New Vault password: 1234

Confirm New Vault password: 1234

It will encrypt your playbook

If you want to see playbook content just use following command

# cat service.yml

Output:

$ANSIBLE\_VAULT;1.1;AES256

31393939306162306439353735383965343764613730323464386463616637303935613634303433

3333376363616565356164646532623535393330326438370a663631333535356631633237333466

33376131643930656662383838663134336434386634333635303232373663653866303365323937

3262336633373765370a393962313736663939373433346265626662383964383961633166383133

37616662646533646233393564613764323639323230363630383763313331623936636434613831

62656666343631303131656165376536646130393861306663636232356464653833353336366433

36393937626339623461333432306162646266336537333262303936343232636234666238316235

31616638336366333863653561336536653237316439396462326363363461323031373235386663

34353733316339613764306237363137326161633739663532393866376539383064356566353431

34373862663131376131343166336130653965613939323830393432386337656465636332343136

336539353134623066366439333661316366

You cvan’t see content for this you need to decrypt your playbook or view it using below command

# ansible-vault view service.yml

Vault password: 123

---

- hosts: all

tasks:

- name: start service

service: name=apache2 state=started

when: ansible\_os\_family == "Debian"

Create encryption for playbook while writing new playbook use below command

# ansible-vault create mybook.yml

New Vault password: 1234

Confirm New Vault password: 1234

---

- hosts: all

tasks:

- name: find hostname

raw: hostname

register: test

- debug="{{ test.stdout }}"

:wq!

# cat mybook.yml

Output:

35656536373434343464623639316331346461313332303563376535383834383664633464376138

3839303337366237666264633933613335366437353130320a326138353765383035333366386230

38316634326139353261383032346433313739643361356566666366336662316135323238336130

3634623637343439350a386632346139306164383462373761393366393533366636666234633832

31373962343538313264393732656132313163626361393166336538653831376663666134646661

64356134336461343264643562323436363861306332356431623036353338396636653262396234

39616536346630383163363364333739623864366439363532303464336531356334303935366539

31383037356564333630326631653461333135386230643133373164383333373632343937623938

35653338386463646564393835653738333538656534393330323837613938616364393061336639

30313064323234613665313738336461626465336366366361666536313232343637376264616530

656430346238363866346665653836323034

Edit the existing playbook use below command

# ansible-valut edit mybook.yml

Vault password: 1234

New Vault password: 1234

Confirm New Vault password: 1234

---

- hosts: all

tasks:

- name: find hostname

raw: hostname

register: tes

- debug="{{ test.stdout }}"

- name: create file

file: path=/tmp/test state=touch

in this playbook we made changes last two lines for creation of file in ‘/tmp’ directory.

Then save and exit using

:wq!

Decrypt the encrypted playbook using below command

# ansible-vault decrypt mybook.yml

Vault password: 1234

Decryption successful

Now check content using cat command

# cat mybook.yml

---

- hosts: all

tasks:

- name: find hostname

raw: hostname

register: test

- debug="{{ test.stdout }}"

- name: create file

file: path=/tmp/test state=touch

**Roles:**

Roles are ways of automatically loading certain vars\_files, tasks, and handlers based on a known file structure. Grouping content by roles also allows easy sharing of roles with other users.

## [Role Directory Structure](http://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html#id5)

Example project structure:

site.yml

webservers.yml

fooservers.yml

roles/

common/

tasks/

handlers/

files/

templates/

vars/

defaults/

meta/

webservers/

tasks/

defaults/

meta/

Roles expect files to be in certain directory names. Roles must include at least one of these directories, however it is perfectly fine to exclude any which are not being used. When in use, each directory must contain a main.yml file, which contains the relevant content:

* tasks - contains the main list of tasks to be executed by the role.
* handlers - contains handlers, which may be used by this role or even anywhere outside this role.
* defaults - default variables for the role (see [Variables](http://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html) for more information).
* vars - other variables for the role (see [Variables](http://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html) for more information).
* files - contains files which can be deployed via this role.
* templates - contains templates which can be deployed via this role.
* meta - defines some meta data for this role. See below for more details.

**Create role using below command:**

**#** cd /etc/ansible/roles

**Requirement:**

Build nginx webserver on redhat family using ansible roles

# ansible-galaxy init webserver

Output:

- webserver was created successfully

# ls

webserver

# cd webserver

# ls

defaults handlers README.md templates vars

files meta tasks tests

# cd files

Downloading nginx rpm package to store static content in files directory

Using below command

# wget <https://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.0-1.el6.ngx.x86_64.rpm>

# vi index.html

<html><body>Welcome to ansible roles</body></html>

After download and creating index.html file just list files dir using below command

# ls

nginx-1.10.0-1.el6.ngx.x86\_64.rpm index.html

step back webserver directory from files

# cd ..

# ls

defaults handlers README.md templates vars

files meta tasks tests

# cd tasks

Here we write code

# ls

Main.yml

# vi main.yml

---

# tasks file for webserver

- name: finding hostname

raw: hostname

register: host

- debug: msg="{{ host.stdout }}"

- name: install nginx on redhat family using rpm pkg

copy: src=nginx-1.10.0-1.el6.ngx.x86\_64.rpm dest=/root

when: ansible\_os\_family == "RedHat" and (host.stdout == "ansi-client\n" or host.stdout == "ansi-client\r\n")

- name: install rpm package for nginx

raw: rpm -ivh nginx-1.10.0-1.el6.ngx.x86\_64.rpm

when: ansible\_os\_family == "RedHat" and (host.stdout == "ansi-client\n" or host.stdout == "ansi-client\r\n")

#- name: create parent directory

# raw: mkdir -p /var/www/html

- name: deploy webserver into clients

copy: src=index.html dest=/usr/share/nginx/html

notify: restart nginx conditionally from handlers

here notify is the module to send notification to handlers if any changes or after deploy the index.html file into clients

:wq!

# cd /etc/ansible/roles/webserver

# ls

defaults handlers README.md templates vars

files meta tasks tests

# cd handlers

Here handlers is to restart nginx service after deploy index.html

# ls

Main.yml

# vi main.yml

---

# handlers file for webserver

- name: restart nginx conditionally from handlers

service: name=nginx state=restarted

when: ansible\_os\_family == "RedHat"

here – name description should be same as notify in tasks to restart services after deploying or find any changes on index.html file.

:wq!

Create playbook to execute roles

# vi /etc/ansible/roles/webserver/execute\_roles.yml

---

- hosts: all

# serial: 1 (if you want run playbook as serially use serial module)

roles:

- webserver

Output:

root@ansi-server:/etc/ansible/roles/webserver# ansible-playbook execute\_role.yml -i hosts.ini

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

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server]

TASK [webserver : finding hostname] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [ansi-server]

TASK [webserver : debug] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-server] => {

"msg": "ansi-server\n"

}

TASK [webserver : install nginx on redhat family using rpm pkg] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

TASK [webserver : install rpm package for nginx] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

TASK [webserver : deploy webserver into clients] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [ansi-server]

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

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-client]

TASK [webserver : finding hostname] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [ansi-client]

TASK [webserver : debug] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-client] => {

"msg": "ansi-client\r\n"

}

TASK [webserver : install nginx on redhat family using rpm pkg] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [ansi-client]

TASK [webserver : install rpm package for nginx] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [ansi-client]

TASK [webserver : deploy webserver into clients] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [ansi-client]

RUNNING HANDLER [webserver : restart nginx conditionally from handlers] \*\*\*\*\*\*\*\*

changed: [ansi-client]

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

ansi-client : ok=7 changed=4 unreachable=0 failed=0

ansi-server : ok=3 changed=1 unreachable=0 failed=0

after ran the playbook goto aws console and open port number as 80 and 443 and put it it as anywhere then take public ip of ec2 instance and hit ip address in your favorite browser

Finally you will get below output from nginx server

