|  |
| --- |
|  |
| Ansible Document |
| Ansible Lab Practice |
|  |
| **Created by : Ananthraj** |
| **2/3/2020** |

|  |
| --- |
|  |

Contents

[1. Ansible Inventory 4](#_Toc517932315)

[1.1 Static Inventory 4](#_Toc517932316)

[1.2 Define the inventory 5](#_Toc517932317)

[1.3 Dynamic Inventory 6](#_Toc517932318)

[2. Privilege Escalation 8](#_Toc517932319)

[3. Ad-hoc Commands 9](#_Toc517932320)

[4. Yaml – Yet another markup language 9](#_Toc517932321)

[5. Variables 13](#_Toc517932322)

[4.1 Variables Defined in Inventory 13](#_Toc517932323)

[4.2 Variables Defined in a Playbook 15](#_Toc517932324)

[4.3 Group\_vars 16](#_Toc517932325)

[4.4 Host-vars 17](#_Toc517932326)

[6. Register Variables 19](#_Toc517932327)

[7. Gather facts 21](#_Toc517932328)

[8. Magic Variables 24](#_Toc517932329)

[9. Managing Inclusion 25](#_Toc517932330)

[8.2 Define the variables in variable.yml and call the variable file in playbook 26](#_Toc517932331)

[8.3 Declare the paths in path variable and get the particular path from fileservers Yaml 27](#_Toc517932332)

[8.4 Calling Outside variables or tasks and include in to a playbook yaml 29](#_Toc517932333)

[10. Nested Loops 30](#_Toc517932334)

[9.1 User creation and group membership using Nested loop with\_items 31](#_Toc517932335)

[9.2 User creation and group membership using Nested loop with\_items 31](#_Toc517932336)

[9.3 Nested if with\_file - Create the below files for testing purpose 33](#_Toc517932337)

[9.4 Nested loop – Fileglobs (with\_fileglobs) 34](#_Toc517932338)

[11. Ansible when Statement 35](#_Toc517932339)

[12. Combining Loops and Conditional Tasks 38](#_Toc517932340)

[13. Handlers 41](#_Toc517932341)

[13.1 Force handlers 42](#_Toc517932342)

[14. Implementing Tags 44](#_Toc517932343)

[15. Error handling 45](#_Toc517932344)

[15.1 Ignore errors 46](#_Toc517932345)

[15.2 Ansible Blocks and Error handling 47](#_Toc517932346)

[16. Overriding The Changed Result 52](#_Toc517932347)

[16.1 changed\_when 52](#_Toc517932348)

[17. Controlling What Defines Failure 54](#_Toc517932349)

[18. Jinja2 Templates 54](#_Toc517932350)

[19. Roles 55](#_Toc517932351)

[19.1 Ansible Galaxy Roles 58](#_Toc517932352)

[20. Selecting Hosts with Host Patterns 61](#_Toc517932353)

[21. Configuring Delegation 63](#_Toc517932354)

[21.1 Delegating tasks to the local machine 63](#_Toc517932355)

[21.2 Delegating a task to a host outside the play 64](#_Toc517932356)

[21.3 Delegating a task to a host that exists in the inventory 65](#_Toc517932357)

[21.4 Delegating a task to a host that does not exist in the inventory 65](#_Toc517932358)

[21.5 Delegated Facts 66](#_Toc517932359)

[22. Configuring Parallelism 68](#_Toc517932360)

[22.1 Configure parallelism in Ansible using forks 68](#_Toc517932361)

[22.2 Running tasks in parallel 68](#_Toc517932362)

[22.3 Asynchronous tasks 69](#_Toc517932363)

[23. Ansible Vault 72](#_Toc517932364)

[23.1 Create an encrypted file 72](#_Toc517932365)

[23.2 Edit an existing encrypted file 73](#_Toc517932366)

[23.3 Change the password for an encrypted file 73](#_Toc517932367)

[23.4 Encrypting an existing file 73](#_Toc517932368)

[23.5 Viewing an encrypted file 74](#_Toc517932369)

[23.6 Decrypting an existing file 74](#_Toc517932370)

[23.7 Encrypting an existing file 74](#_Toc517932371)

# Ansible Inventory

for i in {servera, serverb}

do

ssh root@servera -C hostname

ini live file

yum installed ansible - Check the ansible installed

From controller server

login as student

/etc/ansible/ansible.cfg = ansible invertory file

/home/student/inventory - inventory location

or /etc/ansible/hosts

-i it wont check inventory file

without -i it will check inventory file

chmod 644 inventory

ansible servera.lab.exmaple.com -i inventory --list-hosts

## 1.1 Static Inventory

/home/student/inventory

server1.example.com

192.168.1.10

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[dc1-appservers]

serverc.example.com

serverd.example.com

[dc-servers:children]

dc1-webservers

dc1-appservers

~

ansible dc1-webservers -i inventory --list-hosts

ansible dc1-dbservers -i inventory --list-hosts

ansible 192.168.0.10 -i inventory --list-hosts

ansible dc-servers --list-hosts

ansible dc-servers --list-hosts | more

ansible 192.168.0.204 --list-hosts

ansible 192.168.0.20 --list-hosts

[student@servera ~]$ ansible -i inventory dc1-webservers --list-hosts

hosts (2):

servera.example.com

serverb.example.com

[student@servera ~]$ ansible dc1-webservers --list-hosts

hosts (2):

servera.example.com

serverb.example.com

[student@servera ~]$ ansible 192.168.0.10 --list-hosts

hosts (1):

192.168.0.10

[student@servera ~]$ ansible dc-servers --list-hosts

hosts (4):

servera.example.com

serverb.example.com

serverc.example.com

serverd.example.com

[student@servera ~]$ ansible 192.168.0.20 --list-hosts

hosts (1):

192.168.0.20

[student@servera ~]$ ansible 192.168.0.204 --list-hosts

hosts (1):

192.168.0.204

ansible dc-servers 192.168.0.10 -i inventory --list-hosts

ansible dc-servers 192.168.0.100 -i inventory --list-hosts

ansible dc-servers 192.168.0.255 -i inventory --list-hosts

**Remove -i option**

Note : Once you define the inventory path under / etc/ansible/ansible.cfg , no need to given –I option

vi /etc/ansible/ansible.cfg -> **ansible invertory configuration file**

## 1.2 Define the inventory

inventory = /home/student

inventory = /root/inventory

ansible 192.168.0.100 --list-hosts

ansible inventory dc1-webservers --list-hosts

OR

ansible -i inventory dc1-webservers --list-hosts

Once you update the inventory directory path in /etc/ansible/ansible.cfg file,

we don’t need to give -i option, this inventory will take automatically from the ansible.cfg file

## 1.3 Dynamic Inventory

1) Inventory file should be directory

2) Files that pulled should be executable format

3) The files parsed/analyzed in alphabetical order

inventoryw.py

inventorya.py

hosts

/home/student/dynamic

mkdir inventory

pull the python script

./inventorya.py --list

./inventoryw.py --list

cat hosts

chmod 755 inventorya.py

chmod 755 inventoryw.py

mv inventorya.py ainventorya.py

vi /etc/ansible/ansible.cfg

inventory = /home/student/dynamic/inventory

ansible webservers --list-hosts

ansible -i webservers --list-hosts

ansible -i inventory webservers --list-hosts

We should execute the ansible command from home/student/dynamic

ansible -i inven web --list-hosts -v -> **Can see the verbose mode**

cat /etc/ansible/ansible.cfg | grep '^\[' **-> default parameters list**

#### 1.4 Inventory configuration Methods

1. Default config file = /etc/ansible/ansible.cfg

2. Higher precedence = ~/.ansible.cfg 3. Next higher precedence = current location's ansible.cfg

4. Highest precedence = export global variable ANSIBLE\_CONFIG

**1 .Default Config file** :- The ansible.cfg inventory path should be define in /etc/ansible directory location

**Example : /etc/ansible/ansible.cfg**

2. **Higher precedence: -** The ansible.cfg inventory path should be define in under user home directory

**Example: /home/student/.ansible.cfg**

[student@servera ~]$ cat .ansible.cfg

[defaults]

inventory = /home/student

[student@servera ~]$ cat /home/student/stu-inv

server2.example.com

192.168.1.10

[latam-webservers]

serverlaw1.example.com

serverlaw2.example.com

[latam-appservers]

serverlap1.example.com

serverlap2.example.com

[student@servera ~]$ ansible latam-webservers --list-hosts -v

Using /home/student/.ansible.cfg as config file

hosts (2):

serverlaw1.example.com

serverlaw2.example.com

**3. Next higher precedence:-** The ansible.cfg inventory path should be define in current path/directly location

**Example: /home/student/current**

[root@servera student]# pwd

/home/student/current

[root@servera student]# cat ansible.cfg

[defaults]

inventory = inv1

[root@servera student]# cat inv1

server3.example.com

192.168.2.10

[emia-webservers]

serveremw1.example.com

serveremw2.example.com

[emia-appservers]

serveremap1.example.com

serveremp2.example.com

[student@servera ~]$ ansible emia-webservers --list-hosts -v

Using /home/student/ansible.cfg as config file

hosts (2):

serveremw1.example.com

serveremw2.example.com

[student@servera ~]$ ansible emia-appservers --list-hosts -v

Using /home/student/ansible.cfg as config file

hosts (2):

serveremap1.example.com

serveremp2.example.com

**4. Highest precedence =** The ansible.cfg inventory path should be export in global variable

**Example: export global variable ANSIBLE\_CONFIG**

[student@servera my-dir]$ export ANSIBLE\_CONFIG=/home/student/my-dir

[student@servera my-dir]$ echo $ANSIBLE\_CONFIG

/home/student/my-dir

[student@servera my-dir]$ cd /home/student/my-dir

[student@servera my-dir]$ ls -ltr

total 8

-rw-rw-r--. 1 student student 45 Mar 6 21:18 ansible.cfg

-rw-rw-r--. 1 student student 133 Mar 6 21:18 myinv

[student@servera my-dir]$ cat ansible.cfg

[defaults]

inventory = /home/student/my-dir

[student@servera my-dir]$ cat myinv

[hkg-webservers]

serverhkgw1.example.com

serverhkgw2.example.com

[hkg-appservers]

serverhkgap1.example.com

serverhkgap2.example.com

# Privilege Escalation

User Alias USER - murali

Cmnd alias =

USER ALL=(ALL) CMD

su - murali

sudo useradd u10

[root@ansible ansible]# vi /etc/sudoers.d/ansible

ansible ALL=(ALL) NOPASSWD: ALL

[student@workstation ~]$ cat ansible.cfg

[defaults]

inventory = inventory

**remote\_user=student or devops -. Define the remote user , one user at a time**

ask\_pass=false -> if it’s true means, It will ask the sudo password

[privilege\_escalation]

become=True

become\_method=sudo

become\_user=root

become\_ask\_pass=false - if it’s true means, It will ask become sudo /ssh password

[student@workstation ~]$ cat inventory

server1.example.com

192.168.1.10

[my-servers] **-> my servers setup IP’s**

172.25.250.11

172.25.250.12

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[dc1-appservers]

serverc.example.com

serverd.example.com

[dc-servers:children]

dc1-webservers

dc1-appservers

[root@servera sudoers.d]# pwd

/etc/sudoers.d

[root@servera sudoers.d]# cat devops

devops ALL=(ALL) NOPASSWD: ALL

[root@servera sudoers.d]#

# Ad-hoc Commands

**ansible-doc -l** **-> To view the ansible module details**

**ansible yum** **->**  **Can view the Yum module Parameters**

**ansible-doc user -> Can view User module Parameters**

ansible -m shell servera.lab.example.comansible -m command -a 'ip a' servera.lab.example.comansible -m command -a 'hostname' servera.lab.example.comansible -m command -a 'ip a' servera.lab.example.comansible -m command -a 'ip' servera.lab.example.comansible -m yum -a 'name = httpd state=list' ansible -m command -a 'yum list' servera.lab.example.comansible -m command -a 'yum list| grep zlib' servera.lab.example.comansible -m command -a 'yum list zlib' servera.lab.example.comansible -a 'yum list zlib' servera.lab.example.comansible -a 'ip a' servera.lab.example.comansible -m command -a 'ip a' my-servers –v

cat inventoryansible -a 'ip a' dc1-webserversansible -m ping my-servers

ansible -m user -a 'name=r12 state=remove' servera.lab.example.com -vansible -m user -a 'name=r12 state=absent' servera.lab.example.com -vansible -m user -a 'name=r12 state=absent' servera.lab.example.com -b –v

# Yaml – Yet another markup language

[student@workstation ~]$ **cat test.yml**

---

- name: useradd

# hosts: servera.lab.example.com

tasks:

- name: useradd task

user:

name: t13

state: present

Before apply the playbook

1. Check the inventory and ansible.cfg file
2. Check the yaml mould parameters are correct

**Check the playbook**

ansible-playbook --syntax-check test.yml

ansible-playbook --syntax-check test.yml -C

ansible-playbook test.yml

**Idempotent = Once the changes/execution happened to remote server with mentioned remote users., the same changes/execution or will not overwrite to remote server.**

**ansible-doc –l |grep -i move** – search the move command parameters

**Yaml - Example Coding 1 – Install dialog rpm to servera and serverb**

[student@workstation ~]$ pwd

/home/student

Inventory

[student@workstation ~]$ cat inventory

server1.example.com

192.168.1.10

[my-servers1]

172.25.250.11

172.25.250.12

**[my-servers2]**

**servera**

**serverb**

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

**Ansilbe Config File under /home/student**

[student@workstation ~]$ cat ansible.cfg

[defaults]

**inventory = inventory**

**remote\_user=devops**

**ask\_pass=false**

[privilege\_escalation]

become=True

become\_method=sudo

become\_user=root

**become\_ask\_pass=false**

**Yaml Coding to install dialog RPM**

[student@workstation ~]$ **cat myplay.yml**

---

- name: first play

hosts: my-servers2

tasks:

- name: first task - dialog install

yum:

name: dialog

state: latest

[student@workstation ~]$ **ansible-playbook myplay.yml**

PLAY [first play] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [first task - dialog install] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

changed: [serverb]

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

servera : ok=2 changed=0 unreachable=0 failed=0

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

**Yaml Coding 2 – Multiple tasks Defined**

[student@workstation ~]$ pwd

/home/student

Inventory file

[student@workstation ~]$ cat inventory

server1.example.com

192.168.1.10

[my-servers1]

172.25.250.11

172.25.250.12

**[my-servers2]**

**servera**

**serverb**

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

**[student@workstation ~]$ cat multitsk.yml**

---

- name: Firt play

hosts: **my-servers2 => servera and severb defined in inventory my-server2 group**

tasks:

- name: First Task - Install httpd Pacakge

yum:

name: httpd

state: latest

- name: start http service

service:

name: httpd

state: started

- name: Second Task - Install screen package

yum:

name: screen

state : latest

[student@workstation ~]$ **ansible-playbook --syntax-check multitsk.yml**

playbook: multitsk.yml

[**student@workstation ~]$ ansible-playbook multitsk.yml**

PLAY [Firt play] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Install httpd Pacakge] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

TASK [start http service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

TASK [install screen package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

servera : ok=4 changed=3 unreachable=0 failed=0

serverb : ok=4 changed=3 unreachable=0 failed=0

**Yaml Coding 3 – Multiple Play & tasks Defined**

Inventory file

[student@workstation ~]$ cat inventory

**Localhost**

**workstation**

[my-servers1]

172.25.250.11

172.25.250.12

**[my-servers2]**

**servera**

**serverb**

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

[student@workstation ~]$ **cat multiplay.yml**

---

- name: First Play

hosts: **my-servers2 => servera and severb defined in inventory my-server2 group**

tasks:

- name: First Task - install httpd

yum:

name: httpd

state: latest

- name: copy index.html

copy:

src: index.html

dest: /var/www/html/index.html

mode: 0644

- name: start http service

service:

name: httpd

state: started

enabled: true

- name: second play

hosts: **localhost = it takes localhost**

become: no

tasks:

- name: connect to webserver

uri:

url: http://172.25.250.11

status\_code: 200

# Variables

## 4.1 Variables Defined in Inventory

[student@workstation ~]$ pwd

[student@workstation ~]$ cat inventory

/home/student

localhost

workstation

[my-servers1]

172.25.250.11

172.25.250.12

**[my-servers2]**

**servera**

**server**

**[my-servers2:vars] => Variables Defined for my-servers group to install screen pacakge**

**package=screen**

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

**[student@workstation ~]$ cat inv-variable.yml**

---

- name: Managing vars

hosts: my-servers2

tasks:

- name: Installs the "{{ package }}" package

yum:

name: "{{ package }}"

state: latest

[student@workstation ~]$ **ansible-playbook inv-variable.yml**

PLAY [Managing vars] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Installs the "screen" package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

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

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

## 4.2 [Variables Defined in a Playbook](http://docs.ansible.com/ansible/latest/playbooks_variables.html#id17)

[student@workstation ~]$ **cat inventory**

localhost

workstation

[my-servers1]

172.25.250.11

172.25.250.12

**[my-servers2]**

**servera**

**serverb**

[my-servers2:vars]

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

[student@workstation ~]$ **cat ply-vars.yml**

---

- name: Playbook variables

hosts: my-servers2

vars:

user: joe

tasks:

- name: Creates the user {{ user }}

user:

name: "{{ user }}"

[student@workstation ~]$ **ansible-playbook ply-vars.yml**

PLAY [Playbook variables] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Creates the user joe] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

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

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

## 4.3 Group\_vars

**Create the group\_vars directory current yaml file location**

[student@workstation ~]$ **cat inventory**

localhost

workstation

[my-servers1]

172.25.250.11

172.25.250.12

**[my-servers2]**

**servera**

**serverb**

#[my-servers2:vars]

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

**Group directory creation and define the packages in the group file name**

[student@workstation ~]$ pwd

/home/student

[student@workstation ~]$ mkdir group\_vars

[student@workstation ~]$ cd group\_vars/

[student@workstation group\_vars]$ **cat my-servers2 =. Define the group name**

**package: mariadb -> Define the package name**

[student@workstation ~]$ pwd

/home/student

[student@workstation ~]$ **cat group-vars.yml**

---

- hosts: my-servers2

tasks:

- name: Installs the "{{ package }}" package

yum:

name: "{{ package }}"

state: latest

[student@workstation ~]$ ansible-playbook --syntax-check group-vars.yml

**playbook: group-vars.yml**

[student@workstation ~]$ **ansible-playbook group-vars.yml**

PLAY [my-servers2] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Installs the "mariadb" package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

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

**servera : ok=2 changed=1 unreachable=0 failed=0**

**serverb : ok=2 changed=1 unreachable=0 failed=0**

Note: Here we created **group\_vars** under /home/student/ , the file name **my-servers2** contains the variable package= mariadb .

In this scenario when we execute playbook group**-vars.yml,** the package will be install only on both servers **severa and server.**

## 4.4 Host-vars

**Host directory creation and define the packages in the Inventory server file name**

[student@workstation ~]$ **cat inventory**

localhost

workstation

#servera

#serverb

[my-servers1]

172.25.250.11

172.25.250.12

[my-servers2]

servera

serverb

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

[student@workstation ~]$ pwd

/home/student

[student@workstation ~]$ **mkdir host\_vars**

[student@workstation ~]$ cd host\_vars/

[student@workstation host\_vars]$ **cat servera => Define servera is single server which is available in inventory group name**

**package: telnet => -> Define the package name**

[student@workstation ~]$ **cat group-vars.yml**

---

- name : group or host variable

hosts: **my-servers2 => Define the group name**

tasks:

- name: Installs the "{{ package }}" package

yum:

name: "{{ package }}"

state: latest

[student@workstation ~]$ **ansible-playbook group-vars.yml**

PLAY [group or host variable] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Installs the "telnet" package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [serverb]

changed: [servera]

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

**servera : ok=2 changed=1 unreachable=0 failed=0**

**serverb : ok=2 changed=0 unreachable=0 failed=0**

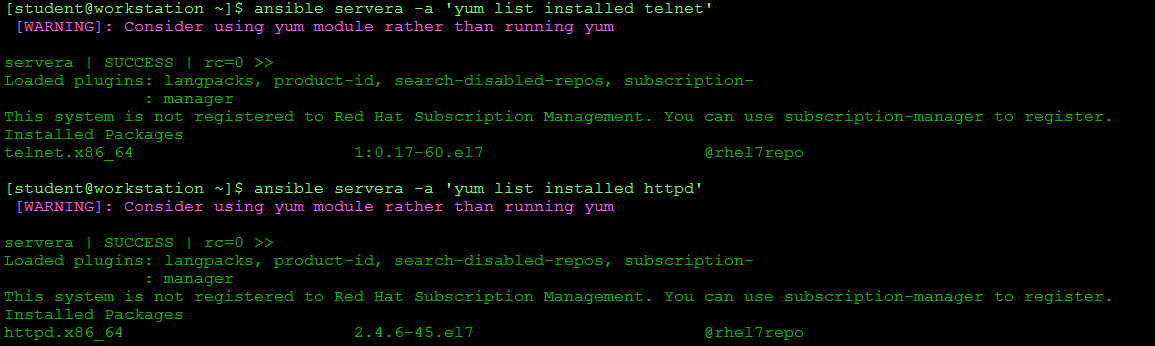
Note: Here we created **host\_vars** under /home/student/ , the file name **servera** contains the variable package= telnet .

In this scenario when we execute playbook group**-vars.yml,** the package will be installonly on **severa** from **my-servers2** group name and does not take **serverb**. Because the **host\_vars is higher precedence**

**Run an ad hoc command to confirm the screen package has been successfully installed.**

[student@workstation ~]$ **ansible servera -a 'yum list installed telnet'**

[student@workstation ~]$ **ansible servera -a 'yum list installed httpd'**

****

**Run the**ansible-playbook**command again, this time using the**-e**option to override the**package**variable.**

[student@workstation ~]$ **ansible-playbook group-vars.yml -e 'package=mutt'**

PLAY [group or host variable] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Installs the "mutt" package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

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

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

# Register Variables

The following playbook demonstrates how to capture the output of a command for debugging purposes

[student@workstation ~]$ **cat inventory**

localhost

workstation

[my-servers1]

172.25.250.11

172.25.250.12

[my-servers2]

servera

serverb

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

[student@workstation ~]$ **cat reg-var.yml**

- name: Register vars

hosts: my-servers2

tasks:

- name: Installs the "{{ package }}" package

yum:

name: "{{ package }}"

state: latest

register: output

- name: collecting output

debug:

output

[student@workstation ~]$ **ansible-playbook reg-var.yml**

PLAY [Facts vars] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Installs the "telnet" package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [serverb]

ok: [servera]

TASK [collecting output] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"output": {

"changed": false,

"failed": false,

"msg": "",

"rc": 0,

"results": [

"All packages providing telnet are up to date",

""

]

}

}

ok: [serverb] => {

"output": {

"changed": false,

"failed": false,

"msg": "",

"rc": 0,

"results": [

"All packages providing mariadb are up to date",

""

]

}

}

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

servera : ok=3 changed=0 unreachable=0 failed=0

serverb : ok=3 changed=0 unreachable=0 failed=0

[student@workstation ~]$ **cat reg2-var.yml**

- name: Installed http packages register details

hosts: my-servers2

tasks:

- name: Install the package

yum:

name: httpd

state: installed

register: install\_result

- name: colleting output

debug:

var: install\_result

**[student@workstation ~]$ ansible-playbook reg2-var.yml**

PLAY [Installed http pacakges register details] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Install the package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [serverb]

ok: [servera]

TASK [colleting output] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"install\_result": {

"changed": false,

"failed": false,

"msg": "",

"rc": 0,

"results": [

**"httpd-2.4.6-45.el7.x86\_64 providing httpd is already installed"**

]

}

}

ok: [serverb] => {

"install\_result": {

"changed": false,

"failed": false,

"msg": "",

"rc": 0,

"results": [

**"httpd-2.4.6-45.el7.x86\_64 providing httpd is already installed"**

]

}

}

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

servera : ok=3 changed=0 unreachable=0 failed=0

serverb : ok=3 changed=0 unreachable=0 failed=0

# Gather facts

Gather all hardware details from the remote servers

| ***Fact*** | ***Variable*** |
| --- | --- |
| Short hostname | **ansible\_hostname** |
| Fully-qualified domain name | **ansible\_fqdn** |
| Main IPv4 address (based on routing) | **ansible\_default\_ipv4.address** |
| A list of the names of all network interfaces | **ansible\_interfaces** |
| Main disk first partition size (based on disk name, such as **vda**, **vdb**, and so on.) | **ansible\_devices.vda.partitions.vda1.size** |
| A list of DNS servers | **ansible\_dns.nameservers** |
| Version of the currently running kernel | **ansible\_kernel** |

**6.1 Facts Filters**

**ansible –m setup servera – Display all facts for remote servers**

**ansible -m setup servera –a 'filter=ansible\_mounts'**

**ansible -m setup servera -a 'filter=ansible\_default\_ipv4'**

**ansible -m setup -a 'filter=ansible\_local' servera**

**ansible -m setup mnode1 -a 'filter=ansible\_fqdn'**

**ansible -m setup mnode1 -a 'filter="ansible\_mounts"'**

**ansible -m setup mnode1 -a 'filter=ansible\_lvm'**

[student@workstation ~]$ **cat inventory**

localhost

workstation

[my-servers1]

172.25.250.11

172.25.250.12

[my-servers2]

servera

serverb

[dc1-webservers]

servera.example.com

serverb.example.com

192.168.0.[1:254]

[servers:children]

my-servers1

my-servers2

[student@workstation ~]$ **cat fact.yml**

---

- name: print ansible facts

hosts: my-servers2

tasks:

- name: Prints various Ansible facts

debug:

msg: >

The dfault ipv4 address of {{ ansible\_fqdn }} is {{ ansible\_default\_ipv4.address }}

[student@workstation ~]$ **ansible-playbook fact.yml**

PLAY [print ansible facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Prints various Ansible facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"msg": "The dfault ipv4 address of servera.exmaple.com is 172.25.250.11 \n"

}

ok: [serverb] => {

"msg": "The dfault ipv4 address of serverb.example.com is 172.25.250.12 \n"

}

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

servera : ok=2 changed=0 unreachable=0 failed=0

serverb : ok=2 changed=0 unreachable=0 failed=0

**6.2 Custom facts - Define the custom facts on servera**

[root@servera facts.d]# pwd

**/etc/ansible/facts.d**

[root@servera facts.d]# **cat mylab.fact**

[packages]

webpkg = httpd

[users]

user = devops

Check the above defined custom facts filter values

student@workstation ~]$ **ansible -m setup servera -a 'filter=ansible\_local'**

servera | SUCCESS => {

"ansible\_facts": {

"ansible\_local": {

"mylab": {

"packages": {

"webpkg": "httpd"

},

"users": {

"user": "devops"

}

}

}

},

"changed": false

}

[student@workstation ~]$ **cat cust-fact.yml**

---

- name : Custom facts play book

hosts: servera

tasks:

- name: Prints various Ansible facts

debug:

msg: >

The package to install on {{ ansible\_fqdn }} is {{ ansible\_local.mylab.packages.webpkg }}

Note : **ansible\_local** – It’s a ansible fact

**mylab** - it’s defined the custom facts file name on servera

**packages – package group defined in mylab**

**webpkg - package name defined in mylab**

[student@workstation ~]$ **ansible-playbook cust-fact.yml**

PLAY [Custom facts play book] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

TASK [Prints various Ansible facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"msg": "The package to install on servera.exmaple.com is httpd\n"

}

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

servera : ok=2 changed=0 unreachable=0 failed=0

# Magic Variables

Some variables are not facts or configured through the **setup** module, but are also automatically set by Ansible. These magic variables can also be useful to get information specific to a particular managed host.

Four of the most useful are:

**hostvars**

Contains the variables for managed hosts, and can be used to get the values for another managed host's variables. It won't include the managed host's facts if they haven't been gathered yet for that host.

**group\_names**

Lists all groups the current managed host is in.

**groups**

Lists all groups and hosts in the inventory.

**inventory\_hostname**

Contains the hostname for the current managed host as configured in the inventory. This may be different from the hostname reported by facts for various reasons.

There are a number of other "magic variables" as well. For more information, see <http://docs.ansible.com/ansible/playbooks_variables.html>. One way to get insight into their values is to use the **debug** module to report on the contents of the **hostvars** variable for a particular host

# Managing Inclusion

When working with complex or long playbooks, administrators can use separate files to divide tasks and lists of variables into smaller pieces for easier management. There are multiple ways to include task files and variables in a playbook.

#### 8.1 Define the tasks in env.yml and call the task file in playbook

[student@workstation ~]$ **cat env.yml**

- name: Installs the {{ package }} package

yum:

name: "{{ package }}"

state: latest

- name: Starts the {{ service }} service

service:

name: "{{ service }}"

state: "{{ state }}"

[student@workstation ~]$ **cat mng1-incl.yml**

---

- name: Install, start, and enable services

hosts: my-servers2

tasks:

- name: Includes the tasks file and defines the variables

include: env.yml

vars:

package: mariadb-server

service: mariadb

state: started

register: output

- name: Debugs the included tasks

debug:

var: output

[student@workstation ~]$ **ansible-playbook mng1-incl.yml**

PLAY [Install, start, and enable services] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Installs the mariadb-server package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

TASK [Starts the mariadb service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

TASK [Debugs the included tasks] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"output": "VARIABLE IS NOT DEFINED!"

}

ok: [serverb] => {

"output": "VARIABLE IS NOT DEFINED!"

}

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

servera : ok=4 changed=2 unreachable=0 failed=0

serverb : ok=4 changed=2 unreachable=0 failed=0

## 8.2 Define the variables in variable.yml and call the variable file in playbook

[student@workstation ~]$ **cat variables.yml**

---

packages:

web\_package: httpd

db\_package: mariadb-server

[student@workstation ~]$ **cat mng2-incl.yml**

---

- name: Install web application packages

hosts: my-servers2

tasks:

- name: Includes the tasks file and defines the variables

include\_vars: variables.yml

- name: Debugs the variables imported

debug:

msg: >

"{{ packages['web\_package'] }} and {{ packages.db\_package }}

have been imported"

[student@workstation ~]$ **ansible-playbook mng-incl2.yml**

PLAY [Install web application packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Includes the tasks file and defines the variables] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

TASK [Debugs the variables imported] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"msg": "\"httpd and mariadb-server have been imported\"\n"

}

ok: [serverb] => {

"msg": "\"httpd and mariadb-server have been imported\"\n"

}

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

servera : ok=3 changed=0 unreachable=0 failed=0

serverb : ok=3 changed=0 unreachable=0 failed=0

## 8.3 Declare the paths in path variable and get the particular path from fileservers Yaml

**Ansible\_fqdn** is a facts it generate fully qualified domain

[student@workstation ~]$ **cat paths.yml**

---

paths:

fileserver: /home/student/srv/filer/{{ ansible\_fqdn }}

dbpath: /home/student/srv/database/{{ ansible\_fqdn }}

[student@workstation ~]$ **cat fileservers.yml**

---

- name: Configure fileservers

hosts: my-servers2

tasks:

- name: Imports the variables file

**include\_vars: paths.yml -> Includes paths variable**

- name: Creates the remote directory

file:

path: "{{ paths.fileserver }}" -> **Define file server path under Path Variable**

state: directory

mode: 0755

register: result

- name: Debugs the results

debug:

var: result

[student@workstation ~]$ **ansible-playbook fileservers.yml**

PLAY [Configure fileservers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Imports the variables file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

TASK [Creates the remote directory] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

TASK [Debugs the results] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"result": {

"changed": true,

"diff": {

"after": {

"path": "/home/student/srv/filer/servera.exmaple.com",

"state": "directory"

},

"before": {

"path": "/home/student/srv/filer/servera.exmaple.com",

"state": "absent"

}

},

"failed": false,

"gid": 0,

"group": "root",

"mode": "0755",

"owner": "root",

"path": "/home/student/srv/filer/servera.exmaple.com",

"secontext": "unconfined\_u:object\_r:user\_home\_t:s0",

"size": 6,

"state": "directory",

"uid": 0

}

}

ok: [serverb] => {

"result": {

"changed": true,

"diff": {

"after": {

"path": "/home/student/srv/filer/serverb.exmaple.com",

"state": "directory"

},

"before": {

"path": "/home/student/srv/filer/serverb.exmaple.com",

"state": "absent"

}

},

"failed": false,

"gid": 0,

"group": "root",

"mode": "0755",

"owner": "root",

"path": "/home/student/srv/filer/serverb.exmaple.com",

"secontext": "unconfined\_u:object\_r:user\_home\_t:s0",

"size": 6,

"state": "directory",

"uid": 0

}

}

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

servera : ok=4 changed=1 unreachable=0 failed=0

serverb : ok=4 changed=1 unreachable=0 failed=0

## 8.4 Calling Outside variables or tasks and include in to a playbook yaml

Variable declaring

[student@workstation ~]$ **cat package.yml**

---

packages:

web\_pkg: httpd

Task Declaration

[student@workstation ~]$ **cat install\_package.yml**

---

- name: Installs {{ packages.web\_pkg }}

yum:

name: "{{ packages.web\_pkg }}"

state: latest

Calling above variable and task in to below playbook

[student@workstation ~]$ **cat bothvars.yml**

---

- name: Install fileserver packages

hosts: my-servers2

tasks:

- name: Includes the variable

**include\_vars: package.yml => Variable calling**

- name: Installs the package

**include: install\_package.yml => task calling**

[student@workstation ~]$ **ansible-playbook bothvars.yml**

PLAY [Install fileserver packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Includes the variable] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

TASK [Installs httpd] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

servera : ok=3 changed=1 unreachable=0 failed=0

serverb : ok=3 changed=1 unreachable=0 failed=0

Note : Update the both vars.yml playbook to override the name of the package to install. Append a vars block to the include statement and define a dictionary to override the name of the package to install.

[student@workstation ~]$ **cat bothvars.yml**

---

- name: Install fileserver packages

hosts: my-servers2

tasks:

- name: Includes the variable

include\_vars: package.yml

- name: Installs the package

include: install\_package.yml

vars:

packages:

web\_pkg: ftp

[student@workstation ~]$ **ansible-playbook bothvars.yml**

PLAY [Install fileserver packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Includes the variable] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

TASK [Installs ftp] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

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

servera : ok=3 changed=1 unreachable=0 failed=0

serverb : ok=3 changed=1 unreachable=0 failed=0

# Nested Loops

| **Loop Keyword** | **Description** |
| --- | --- |
| **with\_file** | Takes a list of control node file names. **item** is set to the content of each file in sequence. |
| **with\_fileglob** | Takes a file name globbing pattern. **item** is set to each file in a directory on the control node that matches that pattern, in sequence, non-recursively. |
| **with\_sequence** | Generates a sequence of items in increasing numerical order. Can take **start** and **end** arguments which have a decimal, octal, or hexadecimal integer value. |
| **with\_random\_choice** | Takes a list. **item** is set to one of the list items at random. |

## 9.1 User creation and group membership using Nested loop with\_items

[student@workstation ~]$ **cat nest1.yml**

---

- name: User creation and group member

hosts: my-servers2

tasks:

- name: Nested if

user:

name: "{{ item[0] }}"

state: present

groups: "{{ item[1] }}"

append: yes

with\_nested:

- [ 'joe', 'jane' ]

- [ 'wheel', 'mysql' ]

[student@workstation ~]$ **ansible-playbook nest1.yml**

PLAY [All DB users have privileges on all databases] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Nested if] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb] => (item=[u'joe', u'wheel'])

changed: [servera] => (item=[u'joe', u'wheel'])

changed: [serverb] => (item=[u'joe', u'mysql'])

changed: [servera] => (item=[u'joe', u'mysql'])

changed: [serverb] => (item=[u'jane', u'wheel'])

changed: [servera] => (item=[u'jane', u'wheel'])

changed: [serverb] => (item=[u'jane', u'mysql'])

changed: [servera] => (item=[u'jane', u'mysql'])

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

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

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

## 9.2 User creation and group membership using Nested loop with\_items

Define the variables **app\_users** and **app\_groups** and call these variable in nested if

[student@workstation ~]$ **cat nest2.yml**

- name: user creation and group member

hosts: my-servers2

vars:

app\_users:

- joe

- jane

- raja

- kumar

app\_groups:

- wheel

- mysql

tasks:

- name: Nested if

user:

name: "{{ item[0] }}"

state: present

groups: "{{ item[1] }}"

append: yes

with\_nested:

- "{{ app\_users }}"

- "{{ app\_groups }}"

[student@workstation ~]$ **ansible-playbook nest2.yml**

PLAY [user creation and group member] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Nested if] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera] => (item=[u'joe', u'wheel'])

changed: [serverb] => (item=[u'joe', u'wheel'])

changed: [serverb] => (item=[u'joe', u'mysql'])

changed: [servera] => (item=[u'joe', u'mysql'])

changed: [servera] => (item=[u'jane', u'wheel'])

changed: [serverb] => (item=[u'jane', u'wheel'])

changed: [serverb] => (item=[u'jane', u'mysql'])

changed: [servera] => (item=[u'jane', u'mysql'])

changed: [serverb] => (item=[u'raja', u'wheel'])

changed: [servera] => (item=[u'raja', u'wheel'])

changed: [servera] => (item=[u'raja', u'mysql'])

changed: [serverb] => (item=[u'raja', u'mysql'])

changed: [serverb] => (item=[u'kumar', u'wheel'])

changed: [servera] => (item=[u'kumar', u'wheel'])

changed: [serverb] => (item=[u'kumar', u'mysql'])

changed: [servera] => (item=[u'kumar', u'mysql'])

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

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

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

## 9.3 Nested if with\_file - Create the below files for testing purpose

/home/student

[student@workstation ~]$ **cat first\_example\_file**

Hello

[student@workstation ~]$ **cat second\_example\_file**

World

[student@workstation ~]$ **cat nest3.yml**

---

- name: Nested if with file

hosts: my-servers2

tasks:

- name: emit a debug message containing the content of each file.

debug:

msg: "{{ item }}"

with\_file:

- **first\_example\_file**

**- second\_example\_file**

[student@workstation ~]$ **ansible-playbook nest3.yml**

PLAY [Nested if with file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [emit a debug message containing the content of each file.] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [serverb] => (item=hello) => {

"changed": false,

"item": "hello",

"msg": "hello"

}

ok: [servera] => (item=hello) => {

"changed": false,

"item": "hello",

"msg": "hello"

}

ok: [servera] => (item=world) => {

"changed": false,

"item": "world",

"msg": "world"

}

ok: [serverb] => (item=world) => {

"changed": false,

"item": "world",

"msg": "world"

}

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

servera : ok=2 changed=0 unreachable=0 failed=0

serverb : ok=2 changed=0 unreachable=0 failed=0

## 9.4 Nested loop – Fileglobs (with\_fileglobs)

mkdir –p /playbooks/files/fooapp/

touch /playbooks/files/fooapp/file1

touch /playbooks/files/fooapp/file2

touch /playbooks/files/fooapp/file3

[student@workstation ~]$ cat file\_globs.yml

---

- name: file globs

hosts: my-servers2

tasks:

- name: Ensure target directory exists

file:

dest: "/etc/fooapp"

state: directory

- name: Copy each file over that matches the given pattern

copy:

src: "{{ item }}"

dest: "/etc/fooapp/"

owner: "root"

mode: 0600

with\_fileglob:

- "/playbooks/files/fooapp/\*"

[student@workstation ~]$ **ansible-playbook file\_globs.yml**

PLAY [file globs] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Ensure target directory exists] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

TASK [Copy each file over that matches the given pattern] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb] => (item=/playbooks/files/fooapp/file1)

changed: [servera] => (item=/playbooks/files/fooapp/file1)

changed: [serverb] => (item=/playbooks/files/fooapp/file2)

changed: [servera] => (item=/playbooks/files/fooapp/file2)

changed: [servera] => (item=/playbooks/files/fooapp/file3)

changed: [serverb] => (item=/playbooks/files/fooapp/file3)

changed: [servera] => (item=/playbooks/files/fooapp/file4)

changed: [serverb] => (item=/playbooks/files/fooapp/file4)

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

servera : ok=3 changed=1 unreachable=0 failed=0

serverb : ok=3 changed=1 unreachable=0 failed=0

# ****Ansible****when****Statement****

One of the simplest conditions which can be tested is whether a Boolean variable is true or false. The **when** statement in the following example causes the task to run only if **run\_my\_task** is true:

**[student@workstation ~]$ cat when1.yml**

**---**

**- name: when condition check**

**hosts: my-servers2**

**vars:**

**run\_my\_task: true**

**tasks:**

**- name: httpd package is installed**

**yum:**

**name: httpd**

**when: run\_my\_task**

[student@workstation ~]$ **ansible-playbook when1.yml**

PLAY [when condition check] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [httpd package is installed] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

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

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

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

The next example is a bit more sophisticated, and tests whether the **my\_service** variable has a value. If it does, the value of **my\_service** is used as the name of the package to install. If the **my\_service** variable is not defined, then the task is skipped without an error.

[student@workstation ~]$ **cat when2.yml**

---

- name: When condition check

hosts: my-servers2

vars:

my\_service: httpd

tasks:

- name: "{{ my\_service }} package is installed"

yum:

name: "{{ my\_service }}"

when: my\_service is defined

[student@workstation ~]$ **ansible-playbook when2.yml**

PLAY [When condition check] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [httpd package is installed] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

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

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

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

In the example for that entry, **my\_special\_user** is a variable which has some value. The variable superusers is a variable which has a list for a value. If the value of **my\_special\_user** is in the superusers list, the conditional passes and the task runs.

[student@workstation ~]$ **cat when3.yml**

---

- name: When condition check

hosts: my-servers2

vars:

my\_special\_user: devops

superusers:

- root

- devops

- toor

tasks:

- name: Task runs if my\_special\_user is in superusers

user:

name: "{{ my\_special\_user }}"

groups: wheel

append: yes

when: my\_special\_user in superusers

[student@workstation ~]$ **ansible-playbook when3.yml**

PLAY [When condition check] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Task runs if my\_special\_user is in superusers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

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

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

Here is another example using the same kind of conditional. It uses two *magic variables* that Ansible automatically sets. The task will only run if the value of the managed host's **inventory\_hostname** variable (containing the name of the managed host from the inventory file) is listed as a member of the host group **my-servers2**

[student@workstation ~]$ **cat when4.yml**

---

- name: Create the database admin

hosts: all

tasks:

- name: user create based on inventory group condition

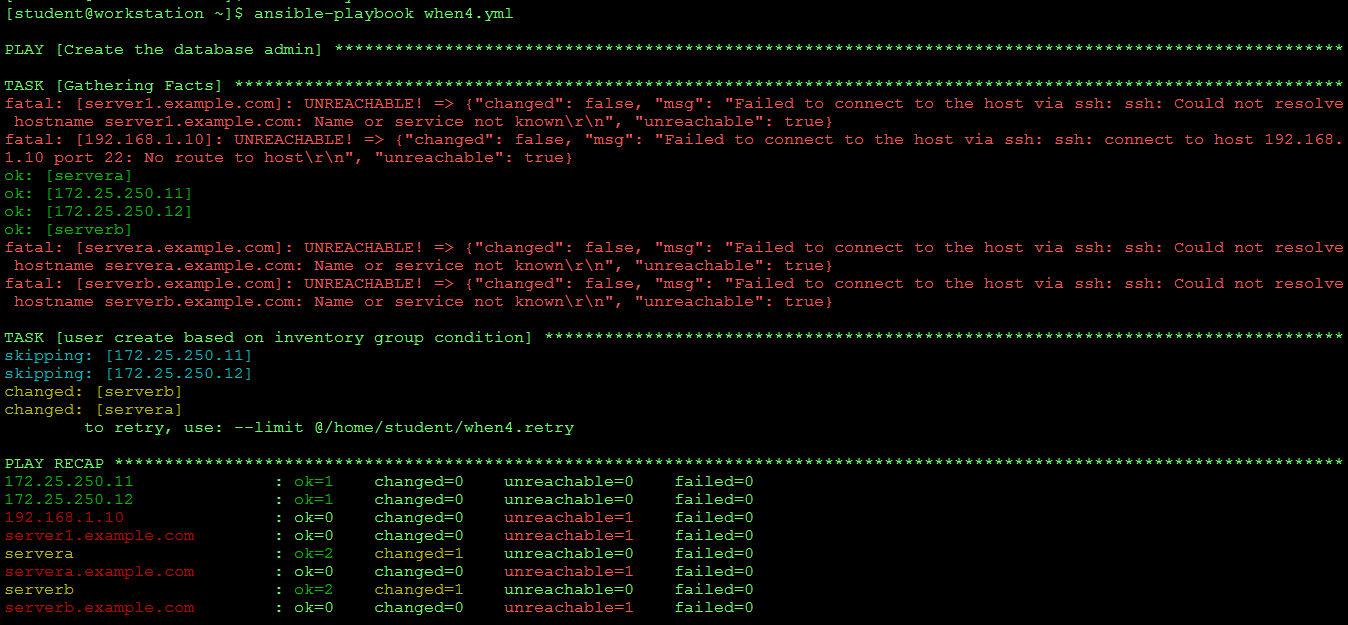
user:

name: db\_admin

state: present

when: inventory\_hostname in groups["my-servers2"]

[student@workstation ~]$ **ansible-playbook when4.yml**



Install the httpd package only the **webservers** **group** servers in the inventory

# Combining Loops and Conditional Tasks

The following annotated playbook will restart the **httpd** service only if the **postfix** service is running.

**[student@workstation ~]$ cat nest-when.yml**

---

- name: Looping and condition checking

hosts: my-servers2

tasks:

- name: Postfix server status

command: /usr/bin/systemctl is-active postfix

ignore\_errors: yes

register: result

- name: result output

debug:

var: result

- name: Restart Apache HTTPD if Postfix running

service:

name: httpd

state: restarted

when: result.rc == 0

student@workstation ~]$ **ansible-playbook nest-when.yml**

PLAY [Looping and condition checking] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Postfix server status] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

TASK [result output] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"result": {

"changed": true,

"cmd": [

"/usr/bin/systemctl",

"is-active",

"postfix"

],

"delta": "0:00:00.017769",

"end": "2018-03-21 13:33:19.189291",

"failed": false,

"rc": 0,

"start": "2018-03-21 13:33:19.171522",

"stderr": "",

"stderr\_lines": [],

"stdout": "active",

"stdout\_lines": [

"active"

]

}

}

ok: [serverb] => {

"result": {

"changed": true,

"cmd": [

"/usr/bin/systemctl",

"is-active",

"postfix"

],

"delta": "0:00:00.016266",

"end": "2018-03-21 13:33:20.444247",

"failed": false,

"rc": 0,

"start": "2018-03-21 13:33:20.427981",

"stderr": "",

"stderr\_lines": [],

"stdout": "active",

"stdout\_lines": [

"active"

]

}

}

TASK [Restart Apache HTTPD if Postfix running] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

servera : ok=4 changed=2 unreachable=0 failed=0

serverb : ok=4 changed=2 unreachable=0 failed=0

The following annotated playbook will not restart the **httpd** service only if the **postfix** service is not running.

[student@workstation ~]$ ansible-playbook nest-when.yml

PLAY [Looping and condition checking] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Postfix server status] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

fatal: [servera]: FAILED! => {"changed": true, "cmd": ["/usr/bin/systemctl", "is-active", "postfix"], "delta": "0:00:00.016629", "end": "2018-03-21 13:35:17.471669", "msg": "non-zero return code", "rc": 3, "start": "2018-03-21 13:35:17.455040", "stderr": "", "stderr\_lines": [], "stdout": "inactive", "stdout\_lines": ["inactive"]}

...ignoring

fatal: [serverb]: FAILED! => {"changed": true, "cmd": ["/usr/bin/systemctl", "is-active", "postfix"], "delta": "0:00:00.019437", "end": "2018-03-21 13:35:18.713628", "msg": "non-zero return code", "rc": 3, "start": "2018-03-21 13:35:18.694191", "stderr": "", "stderr\_lines": [], "stdout": "inactive", "stdout\_lines": ["inactive"]}

...ignoring

TASK [result output] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"result": {

"changed": true,

"cmd": [

"/usr/bin/systemctl",

"is-active",

"postfix"

],

"delta": "0:00:00.016629",

"end": "2018-03-21 13:35:17.471669",

"failed": true,

"msg": "non-zero return code",

"rc": 3,

"start": "2018-03-21 13:35:17.455040",

"stderr": "",

"stderr\_lines": [],

"stdout": "inactive",

"stdout\_lines": [

"inactive"

]

}

}

ok: [serverb] => {

"result": {

"changed": true,

"cmd": [

"/usr/bin/systemctl",

"is-active",

"postfix"

],

"delta": "0:00:00.019437",

"end": "2018-03-21 13:35:18.713628",

"failed": true,

"msg": "non-zero return code",

"rc": 3,

"start": "2018-03-21 13:35:18.694191",

"stderr": "",

"stderr\_lines": [],

"stdout": "inactive",

"stdout\_lines": [

"inactive"

]

}

}

TASK [Restart Apache HTTPD if Postfix running] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

skipping: [serverb]

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

servera : ok=3 changed=1 unreachable=0 failed=0

serverb : ok=3 changed=1 unreachable=0 failed=0

# Handlers

As we’ve mentioned, modules should be idempotent and can relay when they have made a change on the remote system. Playbooks recognize this and have a basic event system that can be used to respond to change.

These ‘notify’ actions are triggered at the end of each block of tasks in a play, and will only be triggered once even if notified by multiple different tasks.

For instance, multiple resources may indicate that apache needs to be restarted because they have changed a config file, but apache will only be bounced once to avoid unnecessary restarts.

[student@workstation ~]$ **cat handler1.yml**

---

- name: Handlers example

hosts: my-servers2

become: true

become\_method: sudo

become\_user: root

tasks:

- name: copy demo.example.conf config template

copy:

src: /var/lib/templates/demo.example.conf.template

dest: /etc/httpd/conf.d/demo.example.bak

notify:

- restart apache

handlers:

- name: restart apache

service:

name: httpd

state: restarted

[student@workstation ~]$ **ansible-playbook handler1.yml**

PLAY [Handlers example] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [copy demo.example.conf config template] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

RUNNING HANDLER [restart apache] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

servera : ok=3 changed=2 unreachable=0 failed=0

serverb : ok=3 changed=2 unreachable=0 failed=0

## Force handlers

When we define any notifier, the handlers will be executed the based on given notifier name.

If any task which **get fails, after given the notifier name**, the **force handlers** will skip the fail task and continue to execute the notifier task

[student@workstation ~]$ **cat handler-force.yml**

---

- name: Using Force handlers

hosts: my-servers2

force\_handlers: yes

become: yes

become\_method: sudo

become\_user: root

tasks:

- name: A task which always notifier its handler

command: /bin/true

notify: restart the database

- name: A task which fails because the package doesn't exist

yum:

name: dialog

state: latest

handlers:

- name: restart the database

service:

name: mariadb

state: restarted

[student@workstation ~]$ **ansible-playbook handler-force.yml**

PLAY [Using Force handlers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [A task which always notifier its handler] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

TASK [A task which fails because the package doesn't exist] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

RUNNING HANDLER [restart the database] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

servera : ok=4 changed=3 unreachable=0 failed=0

serverb : ok=4 changed=3 unreachable=0 failed=0

Below example, we defined **wrong package name : dddialog** in the task after notifier name.

When we execute the play book, the failed task thrown the error and forcefully executed given notifier name.

[student@workstation ~]$ **cat handler-force.yml**

---

- name: Using Force handlers

hosts: my-servers2

force\_handlers: yes

become: yes

become\_method: sudo

become\_user: root

tasks:

- name: A task which always notifier its handler

command: /bin/true

notify: restart the database

- name: A task which fails because the package doesn't exist

yum:

name: dddialog

state: latest

**handlers:**

**- name: restart the database**

service:

name: mariadb

state: restarted

[student@workstation ~]$ **ansible-playbook handler-force.yml**

PLAY [Using Force handlers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [A task which always notifier its handler] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

TASK [A task which fails because the package doesn't exist] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

fatal: [servera]: FAILED! => {"changed": false, "msg": "No package matching 'dddialog' found available, installed or updated", "rc": 126, "results": ["No package matching 'dddialog' found available, installed or updated"]}

fatal: [serverb]: FAILED! => {"changed": false, "msg": "No package matching 'dddialog' found available, installed or updated", "rc": 126, "results": ["No package matching 'dddialog' found available, installed or updated"]}

RUNNING HANDLER [restart the database] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

to retry, use: --limit @/home/student/handler-force.retry

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

servera : ok=3 changed=2 unreachable=0 failed=1

serverb : ok=3 changed=2 unreachable=0 failed=1

# Implementing Tags

If you have a large playbook it may become useful to be able to run a specific part of the configuration without running the whole playbook.

Both plays and tasks support a “tags:” attribute for this reason. You can **ONLY** filter tasks based on tags from the command line with --tags or --skip-tags. Adding “tags:” in any part of a play (including roles) adds those tags to the contained tasks.

[student@workstation ~]$ **cat tags.yml**

---

- name: Example play using tagging

hosts: my-servers2

tasks:

- name: httpd is installed

yum:

name: httpd

state: latest

**tags: webserver**

- name: postfix is installed

yum:

name: postfix

state: latest

[student@workstation ~]$ **ansible-playbook tags.yml --tags 'webserver'**

PLAY [Example play using tagging] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [httpd is installed] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

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

servera : ok=2 changed=0 unreachable=0 failed=0

serverb : ok=2 changed=0 unreachable=0 failed=0

The playbook only ran the task tagged with the **webserver** tag. To **skip tasks with a specific tag and only run the tasks without that tag**, the **--skip-tags** option can be used:

[student@workstation ~]$ **ansible-playbook tags.yml --skip-tags 'webserver'**

PLAY [Example play using tagging] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [postfix is installed] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

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

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

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

# Error handling

We have defined the wrong package name: ddialog in to one task and given correct package: **screen** to in another task. When we execute the playbook, it will failed the both tasks, because we defined the wrong package name, In this situation if you want to ignore the error and continue execute next task, Please use **ignore\_errors: yes**

[student@workstation ~]$ **cat handle-error2.yml**

---

- name: Package installation - nothing

hosts: my-servers2

become: yes

become\_method: sudo

become\_user: root

tasks:

- name: install dialog package

yum:

name: ddialog

state: latest

- name: install screen package

yum:

name: screen

state: latest

[student@workstation ~]$ **ansible-playbook handle-error2.yml**

PLAY [Package installation - nothing] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

TASK [install dialog package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

fatal: [servera]: FAILED! => {"changed": false, "msg": "No package matching 'ddialog' found available, installed or updated", "rc": 126, "results": ["No package matching 'ddialog' found available, installed or updated"]}

to retry, use: --limit @/home/student/handle-error2.retry

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

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

## Ignore errors

We have defined the wrong package name: ddialog in a task and given correct package: **screen** to in to another task. When we execute the playbook, it’s ignoring wrong package and successfully executed next task: **screen** package installed.

[student@workstation ~]$ **cat handle-error2.yml**

---

- name: Package installation - nothing

hosts: my-servers2

become: yes

become\_method: sudo

become\_user: root

**ignore\_errors: yes**

tasks:

- name: install dialog package

yum:

name: ddialog

state: latest

- name: install screen package

yum:

name: screen

state: latest

[student@workstation ~]$ **ansible-playbook handle-error2.yml**

PLAY [Package installation - nothing] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

TASK [install dialog package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

fatal: [servera]: FAILED! => {"changed": false, "msg": "No package matching 'ddialog' found available, installed or updated", "rc": 126, "results": ["No package matching 'ddialog' found available, installed or updated"]}

...ignoring

TASK [install screen package] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

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

servera : ok=3 changed=0 unreachable=0 failed=0

## Ansible Blocks and Error handling

In playbooks, blocks are clauses that logically group tasks, and can be used to control how tasks are executed. For example, a task **block** can have a **when** directive to apply a conditional to multiple tasks

Below find the Inventory the group web-servers contains the serverb. The below play book will be executed multiple tasks(as mentioned **block)** only for **web\_servers group**

[student@workstation ~]$ **cat inventory**

server1.example.com

192.168.1.10

[web-servers]

serverb

[my-servers2]

servera

[dc1-webservers]

servera.example.com

serverb.example.com

#192.168.0.[1:254]

#[servers:children]

#my-servers1

#my-servers2

[student@workstation ~]$ **cat blocks-test.yml**

---

- name: test servera.lab.example.com

hosts: all

become: yes

become\_method: sudo

become\_user: root

tasks:

- block:

- name: installhttp

yum:

name: httpd

state: latest

- name: copy file

copy:

src: httpd.conf

dest: /etc/httpd/conf.d/oselabs.conf

mode: 0644

when: inventory\_hostname in groups['web-servers']

[student@workstation ~]$ **ansible-playbook blocks-test.yml**

PLAY [test servera.lab.example.com] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

fatal: [servera.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname servera.example.com: Name or service not known\r\n", "unreachable": true}

fatal: [server1.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname server1.example.com: Name or service not known\r\n", "unreachable": true}

fatal: [serverb.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname serverb.example.com: Name or service not known\r\n", "unreachable": true}

ok: [servera]

ok: [serverb]

fatal: [192.168.1.10]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.1.10 port 22: No route to host\r\n", "unreachable": true}

TASK [installhttp] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

changed: [serverb]

TASK [copy file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

changed: [serverb]

to retry, use: --limit @/home/student/blocks-test.retry

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

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

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

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

servera.example.com : ok=0 changed=0 unreachable=1 failed=0

serverb : ok=3 changed=2 unreachable=0 failed=0

serverb.example.com : ok=0 changed=0 unreachable=1 failed=0

Below find the Inventory the group web-servers contains the serverb. The below play book will be executed multiple tasks(as mentioned **block)** only for **web\_servers group**

[student@workstation ~]$ **cat blocks2-test.yml**

- name: Install the http only to webservers group

hosts: all

become: yes

become\_method: sudo

become\_user: root

ignore\_errors: yes

tasks:

**- block:**

- name: install httpd

yum:

name: httpd

state: latest

- name: copy the conf file

copy:

src: /etc/httpd/conf/httpd.conf

dest: /etc/httpd/conf/httpd\_bak.conf

mode: 0644

- name: Start the service

service:

name: httpd

state: restarted

when: inventory\_hostname in groups['web-servers']

[student@workstation ~]$ **ansible-playbook blocks2-test.yml**

PLAY [Install the http only to webservers group] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

fatal: [server1.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname server1.example.com: Name or service not known\r\n", "unreachable": true}

fatal: [servera.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname servera.example.com: Name or service not known\r\n", "unreachable": true}

fatal: [serverb.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname serverb.example.com: Name or service not known\r\n", "unreachable": true}

ok: [serverb]

ok: [servera]

fatal: [192.168.1.10]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.1.10 port 22: No route to host\r\n", "unreachable": true}

TASK [install httpd] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

changed: [serverb]

TASK [copy the conf file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

changed: [serverb]

TASK [Start the service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

changed: [serverb]

to retry, use: --limit @/home/student/blocks2-test.retry

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

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

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

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

servera.example.com : ok=0 changed=0 unreachable=1 failed=0

serverb : ok=4 changed=3 unreachable=0 failed=0

serverb.example.com : ok=0 changed=0 unreachable=1 failed=0

**Block, Rescue and Always**

The **rescue** and **always** statements. If any task in a **block** fails, tasks in its **rescue** block are executed in order to recover. After the tasks in the block and possibly the rescue run, then tasks in its **always** block run. To summarize:

* **block**: Defines the main tasks to run.
* **rescue**: Defines the tasks that will be run if the tasks defined in the **block** clause fails.
* **always**: Defines the tasks that will always run independently of the success or failure of tasks defined in the **block** and **rescue** clauses.

The following example shows how to implement a block in a playbook. Even if tasks defined in the **block** clause fail, tasks defined in the **rescue** and **always** clauses will be executed.

tasks:

- block:

- name: upgrade the database

shell:

cmd: /usr/local/lib/upgrade-database

rescue:

- name: revert the database upgrade

shell:

cmd: /usr/local/lib/revert-database

always:

- name: always restart the database

service:

name: mariadb

state: restarted

The **when** condition on a **block** also applies to its **rescue** and **always** sections if present.

Execute the multiple tasks using Block and if any task get failed situation, it will revert back the tasks

student@workstation ~]$ **cat block-rescue.yml**

- name: Blocks Resuce always

hosts: all

become: yes

become\_method: sudo

become\_user: root

tasks:

- block:

- name: install http

yum:

name: httpd

state: latest

- name: copy file

copy:

src: httpd.conf

dest: /tmp/oselabs.conf

mode: 0644

when: inventory\_hostname in groups["web-servers"]

- name: Start the http service

service :

name : httpppdd

state : restarted

rescue:

- name: rescue task

command: echo "resuced"

always:

- name: restart apache

service:

name: httpd

state: restarted

[student@workstation ~]$ **ansible-playbook block-rescue.yml**

PLAY [Blocks Resuce always] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

fatal: [server1.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname server1.example.com: Name or service not known\r\n", "unreachable": true}

fatal: [servera.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname servera.example.com: Name or service not known\r\n", "unreachable": true}

fatal: [serverb.example.com]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname serverb.example.com: Name or service not known\r\n", "unreachable": true}

ok: [serverb]

ok: [servera]

fatal: [192.168.1.10]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.1.10 port 22: No route to host\r\n", "unreachable": true}

TASK [install http] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

changed: [serverb]

TASK [copy file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera]

changed: [serverb]

TASK [Start the http service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

fatal: [serverb]: FAILED! => {"changed": false, "msg": "Unsupported parameters for (systemd) module: always,rescue Supported parameters include: daemon\_reload,enabled,masked,name,no\_block,state,user"}

fatal: [servera]: FAILED! => {"changed": false, "msg": "Unsupported parameters for (systemd) module: always,rescue Supported parameters include: daemon\_reload,enabled,masked,name,no\_block,state,user"}

to retry, use: --limit @/home/student/block-rescue.retry

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

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

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

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

servera.example.com : ok=0 changed=0 unreachable=1 failed=0

**serverb : ok=3 changed=2 unreachable=0 failed=1**

serverb.example.com : ok=0 changed=0 unreachable=1 failed=0

# [Overriding The Changed Result](http://docs.ansible.com/ansible/latest/playbooks_error_handling.html#id9)

When a shell/command or other module runs it will typically report “changed” status based on whether it thinks it affected machine state.

Sometimes you will know, based on the return code or output that it did not make any changes, and wish to override the “changed” result such that it does not appear in report output or does not cause handlers to fire:

Below example shows, when we execute the playbook, will get the register output is welcome then the notifier and handler task executes until when changed register output. When we change register output, the changed result regit

## changed\_when

[root@servera ~]# **cat /usr/local/bin/test.sh**

#! /bin/bash

for i in {1}

do

echo "welcome"

sleep 1

done

[root@serverb ~]# **cat /usr/local/bin/test.sh**

#! /bin/bash

for i in {1}

do

echo "welcome"

sleep 1

done

[student@workstation ~]$ **cat when-changed.yml**

- name: handling errors

hosts: my-servers2

become: true

become\_method: sudo

become\_user: root

tasks:

- name: testing script

shell: /usr/local/bin/test.sh

register: script\_output

changed\_when: "'welcome' in script\_output.stdout" => it will override

Task until change

Different value/string

changed\_when: "'Success' in script\_output.stdout => it will not override

task, because we

changed the string as

success

notify:

- restart apache

- name:

debug:

var: script\_output

handlers:

- name: restart apache

service:

name: httpd

state: restarted

[student@workstation ~]$ **ansible-playbook when-changed.yml**

PLAY [handling errors] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [testing script] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

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

ok: [servera] => {

"script\_output": {

"changed": false,

"cmd": "/usr/local/bin/test.sh",

"delta": "0:00:01.013057",

"end": "2018-03-27 15:23:36.109416",

"failed": false,

"rc": 0,

"start": "2018-03-27 15:23:35.096359",

"stderr": "",

"stderr\_lines": [],

"stdout": "welcome",

"stdout\_lines": [

"welcome"

]

}

}

ok: [serverb] => {

"script\_output": {

"changed": false,

"cmd": "/usr/local/bin/test.sh",

"delta": "0:00:01.013525",

"end": "2018-03-27 15:23:36.143660",

"failed": false,

"rc": 0,

"start": "2018-03-27 15:23:35.130135",

"stderr": "",

"stderr\_lines": [],

"stdout": "welcome",

"stdout\_lines": [

"welcome"

]

}

}

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

servera : ok=3 changed=0 unreachable=0 failed=0

serverb : ok=3 changed=0 unreachable=0 failed=0

# [Controlling What Defines Failure](http://docs.ansible.com/ansible/latest/playbooks_error_handling.html#id8)

Suppose the error code of a command is meaningless and to tell if there is a failure what really matters is the output of the command, for instance if the string “FAILED” is in the output.

# Jinja2 Templates

Ansible uses Jinja2 templating to enable dynamic expressions and access to variables. Ansible greatly expands the number of filters and tests available, as well as adding a new plugin type: lookups.

Ansible uses the Jinja2 templating system to modify files before they are distributed to managed hosts.

[student@workstation ~]$ **cat motd.j2**

This is the system {{ ansible\_hostname }}

Today's date is: {{ ansible\_date\_time.date }}

Only use this system with permission

You can ask {{ system\_owner }} for access

[student@workstation ~]$ cat motd.j2

This is the system {{ ansible\_hostname }}

Today's date is: {{ ansible\_date\_time.date }}

Only use this system with permission

You can ask {{ system\_owner }} for access

[student@workstation ~]$ **cat jinja.yml**

- name: Jinja2 templates

hosts: my-servers2

user: devops

become: true

vars:

system\_owner: clyde@example.com

tasks:

- name: Define Templates

template:

src: motd.j2

dest: /etc/motd

owner: root

group: root

mode: 0644

[student@workstation ~]$ **ansible-playbook jinja.yml**

PLAY [Jinja2 templates] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [Define Templates] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera]

changed: [serverb]

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

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

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

[root@servera bin]# **cat /etc/motd**

This is the system servera

Today's date is: 2018-03-27

Only use this system with permission

You can ask clyde@example.com for access

[root@serverb bin]# **cat /etc/motd**

This is the system serverb

Today's date is: 2018-03-27

Only use this system with permission

You can ask clyde@example.com for access

# Roles

Data centers have a variety of different types of hosts. Some serve as web servers, others as database servers, and others can have software development tools installed and configured on them. An Ansible playbook, with tasks and handlers to handle all of these cases, would become large and complex over time. Ansible roles allow administrators to organize their playbooks into separate, smaller playbooks and files.

Roles provide Ansible with a way to load tasks, handlers, and variables from external files. Static files and templates can also be associated and referenced by a role. The files that define a role have specific names and are organized in a rigid directory structure, which will be discussed later. Roles can be written so they are general purpose and can be reused.

Use of Ansible roles has the following benefits:

* Roles group content, allowing easy sharing of code with others
* Roles can be written that define the essential elements of a system type: web server, database server, git repository, or other purpose
* Roles make larger projects more manageable
* Roles can be developed in parallel by different administrators

Create the role motd under roles directory and also create motd roles Tasks, templates and defaults under roles/motd

[student@workstation motd]$ pwd

/home/student/roles/motd

[student@workstation motd]$ ls -ltr

total 0

drwxrwxr-x. 2 student student 21 Mar 27 16:10 defaults

drwxrwxr-x. 2 student student 21 Mar 27 16:12 templates

drwxrwxr-x. 2 student student 26 Mar 27 16:16 tasks

**Motd Role Task**

[student@workstation tasks]$ pwd

**/home/student/roles/motd/tasks**

[student@workstation tasks]$ **cat main.yml**

---

- name: deliver motd file

template:

src: templates/motd.j2

dest: /etc/motd

owner: root

group: root

mode: 0444

**Motd Defaults**

[student@workstation defaults]$ pwd

/home/student/roles/motd/defaults

[student@workstation defaults]$ **cat main.yml**

---

system\_owner: user@host.example.com

[student@workstation templates]$ pwd

/home/student/roles/motd/templates

[student@workstation templates]$ **cat motd.j2**

This is the system {{ ansible\_hostname }}

Today's date is: {{ ansible\_date\_time.date }}

Only use this system with permission

You can ask {{ system\_owner }} for access

Once defined above **tasks, defaults and templates under role directory,** we have to write playbook to declare the role name

[student@workstation ~]$ pwd

/home/student

[student@workstation ~]$ cat use-motd-role.yml

---

- name: use motd role playbook

hosts: my-servers2

user: devops

become: true

roles: => Define the roles

- motd => Define the role name

[student@workstation ~]$ **ansible-playbook use-motd-role.yml**

PLAY [use motd role playbook] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [motd : deliver motd file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

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

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

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

**[root@servera bin]# cat /etc/motd**

This is the system servera

Today's date is: 2018-03-27

Only use this system with permission

You can ask user@host.example.com for access

**[root@serverb bin]# cat /etc/motd**

This is the system serverb

Today's date is: 2018-03-27

Only use this system with permission

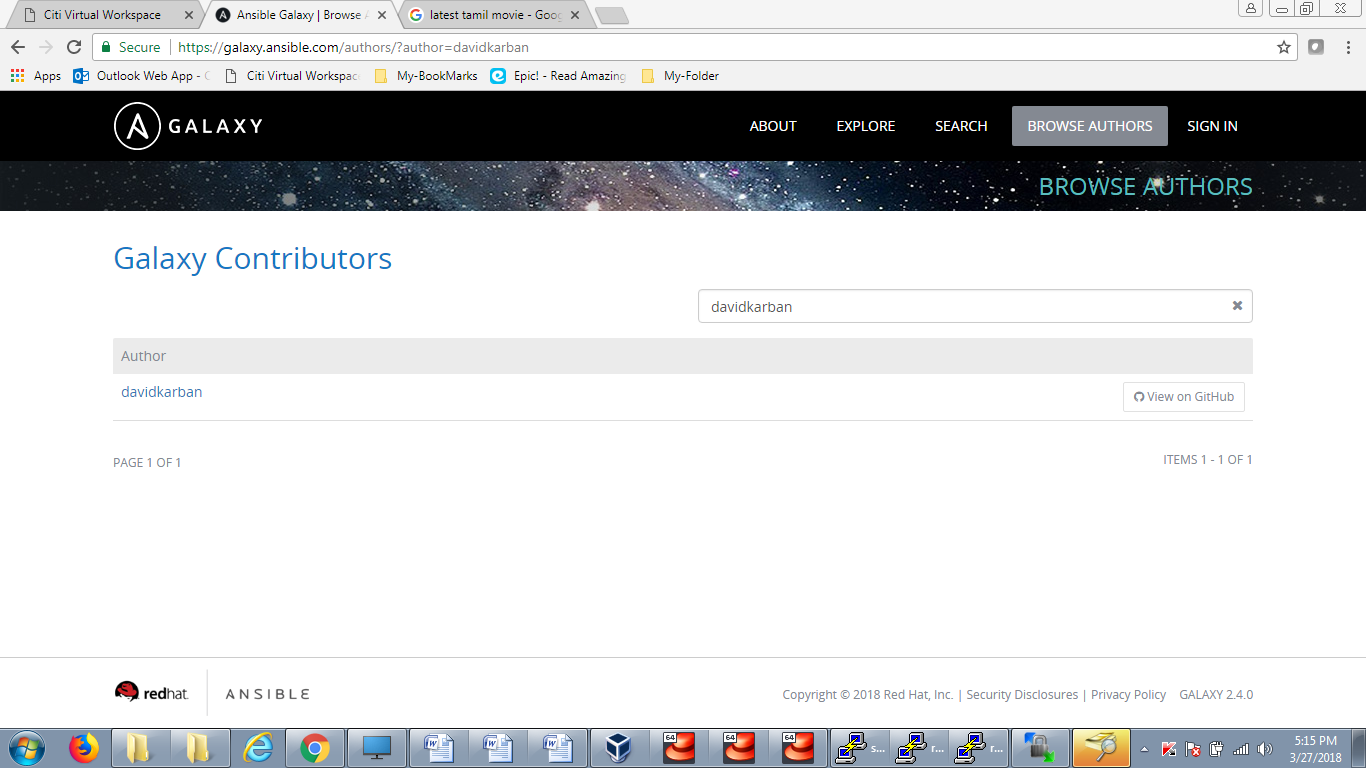
You can ask user@host.example.com for access

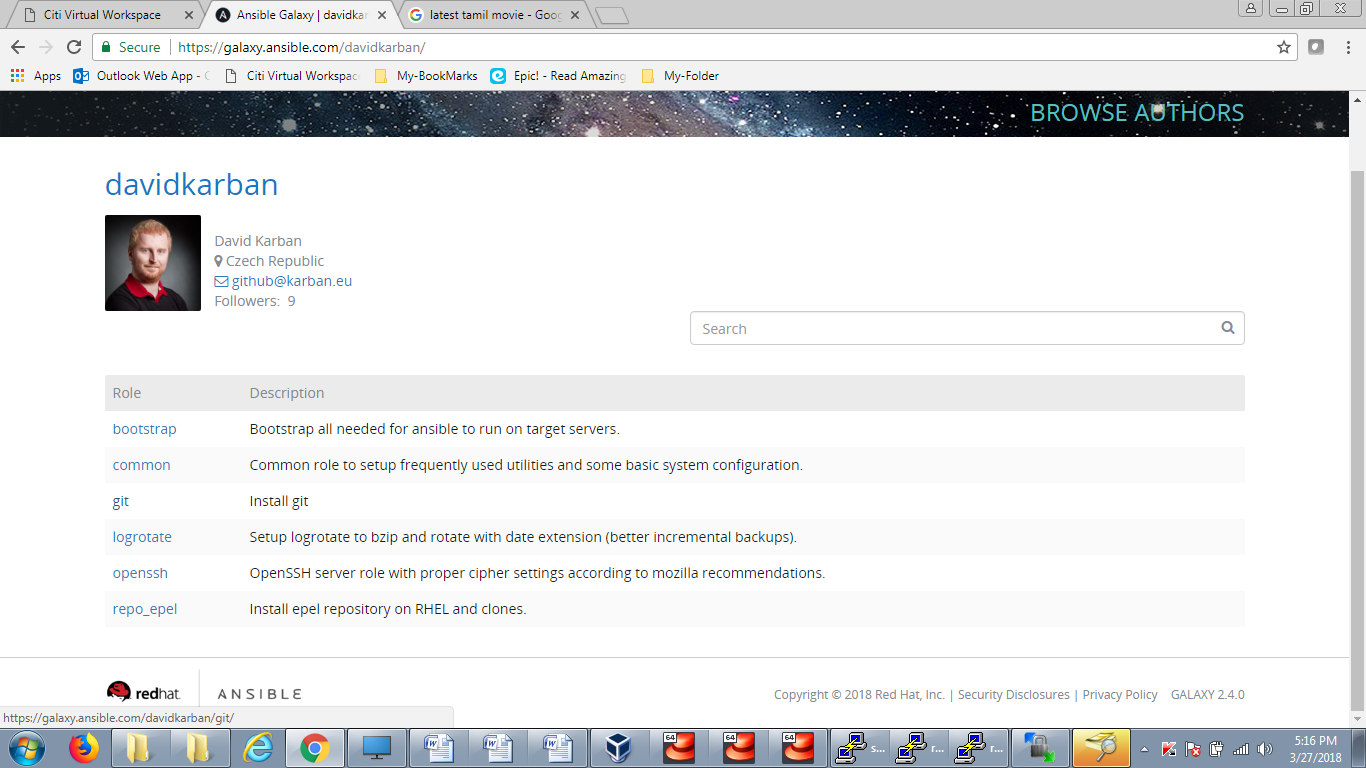
## Ansible Galaxy Roles

[**https://galaxy.ansible.com/davidkarban/git/**](https://galaxy.ansible.com/davidkarban/git/)

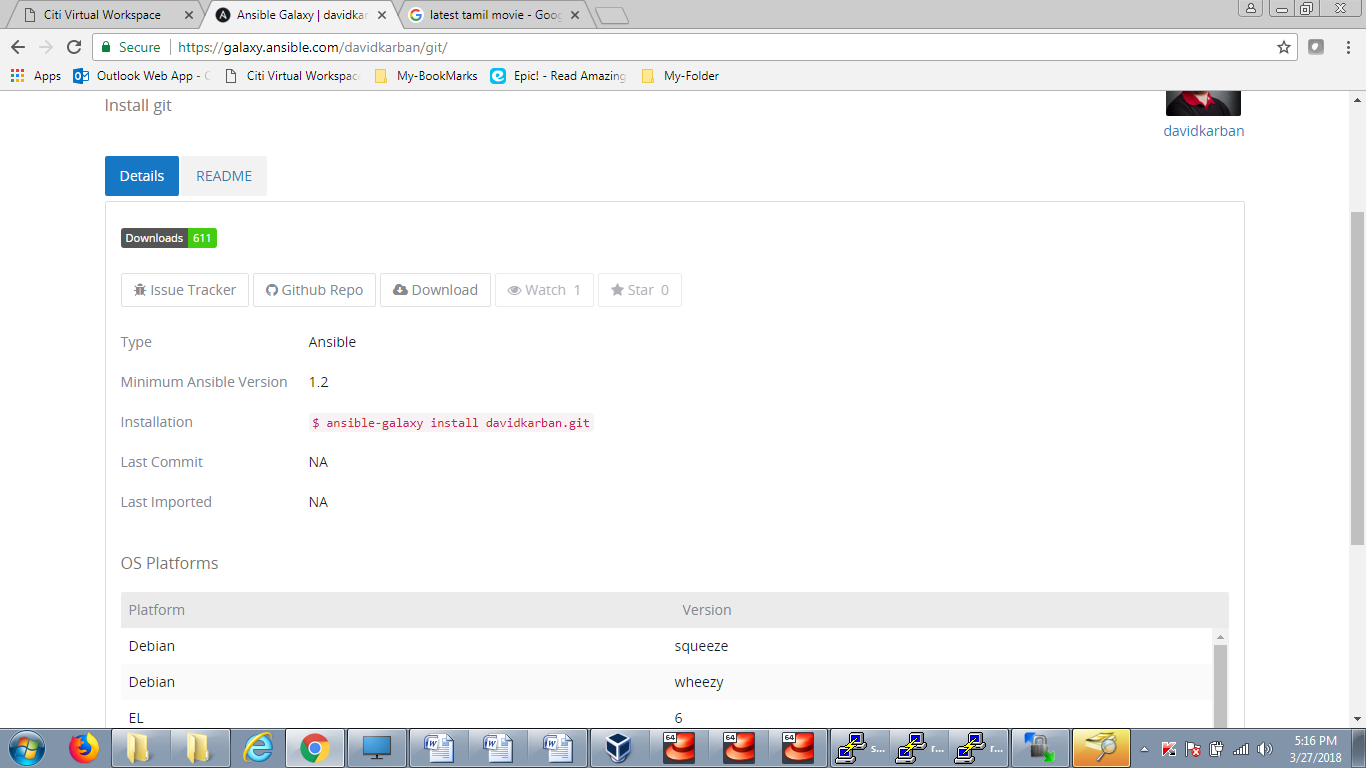
Download the davidharden git author

[**https://galaxy.ansible.com/authors/**](https://galaxy.ansible.com/authors/)

****

****

**Select git and download**

****

Once downloaded and extract the git under roles/davidkarban.git. Below can see **tasks,defaults,handlers ,vars** are defined under roles/<roles name> directory location

[student@workstation davidkarban.git]$ pwd

/home/student/roles/davidkarban.git

[student@workstation davidkarban.git]$ ls -tlr

total 4

-rw-r--r--. 1 student student 1328 May 30 2016 README.md

drwxr-xr-x. 2 student student 22 Mar 27 16:27 defaults

drwxr-xr-x. 2 student student 22 Mar 27 16:27 handlers

drwxr-xr-x. 2 student student 22 Mar 27 16:27 meta

drwxr-xr-x. 2 student student 22 Mar 27 16:27 tasks

drwxr-xr-x. 2 student student 58 Mar 27 16:27 vars

**Can see the tree view directory structure**

[student@workstation davidkarban.git]$ pwd

/home/student/roles/davidkarban.git

[student@workstation davidkarban.git]$ tree

.

├── defaults

│   └── main.yml

├── handlers

│   └── main.yml

├── meta

│   └── main.yml

├── README.md

├── tasks

│   └── main.yml

└── vars

├── Debian.yml

├── main.yml

└── RedHat.yml

5 directories, 8 files

**Roles play book yaml file**

[student@workstation ~]$ pwd

/home/student

[student@workstation ~]$ **cat use-git-role.yml**

---

- name: use davidkarban.git role playbook

hosts: my-servers2

user: devops

become: true

roles:

- davidkarban.git

[student@workstation ~]$ **ansible-playbook use-git-role.yml**

PLAY [use davidkarban.git role playbook] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [davidkarban.git : Load the OS specific variables] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

ok: [serverb]

TASK [davidkarban.git : Install the packages in Redhat derivates] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera] => (item=[u'git'])

changed: [serverb] => (item=[u'git'])

TASK [davidkarban.git : Install the packages in Debian derivates] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [servera] => (item=[])

skipping: [serverb] => (item=[])

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

servera : ok=3 **changed=1** unreachable=0 failed=0

serverb : ok=3 **changed=1** unreachable=0 failed=0

# Selecting Hosts with Host Patterns

List all hosts in the inventory

[student@workstation ~]$ **ansible all --list-hosts**

hosts (8):

server1.example.com

server2.example.com

192.168.1.10

192.168.1.11

servera

serverb

servera.example.com

serverb.example.com

List ungrouped hosts in the inventory

[student@workstation ~]$ **ansible ungrouped --list-hosts**

hosts (4):

server1.example.com

server2.example.com

192.168.1.10

192.168.1.11

List the hosts using wildcards in the inventory

[student@workstation ~]$ **ansible '\*' --list-hosts**

hosts (8):

server1.example.com

server2.example.com

192.168.1.10

192.168.1.11

servera

serverb

servera.example.com

serverb.example.com

List the hosts only example.com the inventory

[student@workstation ~]$ **ansible '\*.example.com' --list-hosts**

hosts (4):

server2.example.com

servera.example.com

serverb.example.com

server1.example.com

List the hosts starting from 192.168.1.\* the inventory

[student@workstation ~]$ **ansible '192.168.1.\*' --list-hosts**

hosts (2):

192.168.1.10

192.168.1.11

List the hosts starting from server\* the inventory

[student@workstation ~]$ **ansible 'server\*' --list-hosts**

hosts (6):

server2.example.com

servera.example.com

serverb.example.com

server1.example.com

servera

server

List the hosts with multiple entries

ansible servera.example.com,serverb.example.com,192.168.1.10 --list-hosts

hosts (3):

servera.example.com

serverb.example.com

192.168.1.10

For example, based on our example inventory, the following host pattern will match machines in group web-server only if they are also in group my-servers2

[student@workstation ~]$ **ansible 'web-servers,&my-servers2' --list-hosts**

hosts (1):

server

This example, given our test inventory, matches all hosts defined in the my-servers2 group with the exception of serverb

[student@workstation ~]$ **ansible 'my-servers2,!serverb' --list-hosts**

hosts (1):

servera

# Configuring Delegation

Delegation can help by performing necessary actions for tasks on hosts other than the managed host being targeted by the play in the inventory. Some scenarios that delegation can handle include:

1. Delegating a task to the local machine
2. Delegating a task to a host outside the play
3. Delegating a task to a host that exists in the inventory
4. Delegating a task to a host that does not exist in the inventory

## Delegating tasks to the local machine

When any action needs to be performed on the node running Ansible, it can be delegated to localhost by using **delegate\_to: localhost**.

Here is a sample playbook which, for each managed host, runs the command **ps** on the managed host and then on **localhost** by using the **delegate\_to** keyword. It displays the output from both command tasks using the debug module. Both tasks which run the command module set **changed\_when:** false because they don't change the state of either system when they run.

[student@workstation ~]$ **cat delegate-local.yml**

---

- name: delegate\_to:localhost example

hosts: servera

tasks:

- name: Get information on managed host processes

command: ps

register: remote\_process

changed\_when: false

- name: Display information on managed host processes

debug:

msg: "{{ remote\_process.stdout }}"

- name: Get information about localhost processes

command: ps

delegate\_to: 172.25.250.10

register: local\_process

changed\_when: false

- name: Display information on localhost processes

debug:

msg: "{{ local\_process.stdout }}"

[[student@workstation ~]$ **ansible-playbook delegate-local.yml**

PLAY [delegate\_to:localhost example] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

TASK [Get information on managed host processes] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera]

TASK [Display information on managed host processes] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"msg": " PID TTY TIME CMD\n 3651 pts/1 00:00:00 sudo\n 3652 pts/1 00:00:00 sh\n 3653 pts/1 00:00:00 python\n 3654 pts/1 00:00:00 python\n 3659 pts/1 00:00:00 ps"

}

TASK [Get information about localhost processes] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera -> 172.25.250.10]

TASK [Display information on localhost processes] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"msg": " PID TTY TIME CMD\n 4510 pts/2 00:00:00 sudo\n 4511 pts/2 00:00:00 sh\n 4512 pts/2 00:00:00 python\n 4513 pts/2 00:00:00 python\n 4518 pts/2 00:00:00 ps"

}

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

servera : ok=5 changed=0 unreachable=0 failed=0

## Delegating a task to a host outside the play

The following example shows Ansible code that will delegate a task to an outside machine serverb.

[student@workstation ~]$ **cat delegate-ouside.yml**

---

- name: Delicate testing Outside server

hosts: servera

tasks:

- name: testing host outside Play

command: hostname {{ inventory\_hostname }}

delegate\_to: serverb

- name: Install vsftp Latest

yum:

name: vsftpd

state: latest

delegate\_to: serverb

- name: Install httpd latest

yum:

name: httpd

state: latest

delegate\_to: serverb

[student@workstation ~]$ **ansible-playbook delegate-ouside.yml**

PLAY [Delicate testing Outside server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

TASK [testing host outside Play] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera -> servera]

TASK [Install vsftp Latest] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera -> serverb]

TASK [Install httpd latest] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera -> serverb]

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

servera : ok=4 changed=1 unreachable=0 failed=0

## Delegating a task to a host that exists in the inventory

When delegating to a host listed in the inventory, the inventory data will be used when creating the connection to the delegation target. This would include settings for **ansible\_connection, ansible\_host, ansible\_port, ansible\_user** and so on. Only the connection-related variables are used; the rest are read from the managed host originally targeted.

## Delegating a task to a host that does not exist in the inventory

When delegating a task to a host that does not exist in the inventory, Ansible will use the same connection type and details used for the managed host to connect to the delegating host. To adjust the connection details, use the add\_host module to create an ephemeral host in your inventory with connection data defined.

---

- name: Delegating a task to a host that does not exist in the inventory

hosts: workstation

tasks:

- name: add Delegation host

add\_host:

name : serverb

ansible\_host: serverb

ansible\_user : devops

- name: echo Hello

command: echo "Hello from {{ inventory\_hostname }}"

#command: touch /tmp/01April

#command: ifconfig

#command: echo "Hello from {{ ansible\_hostname }}"

delegate\_to: serverb

register: output

- name:

debug:

msg: "{{ output.stdout }}"

[student@workstation ~]$ **ansible-playbook delegate-noinv-host.yml -v**

Using /home/student/ansible.cfg as config file

PLAY [Delegating a task to a host that does not exist in the inventory] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [workstation]

TASK [add Delegation host] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [workstation] => {"add\_host": {"groups": [], "host\_name": "serverb", "host\_vars": {"ansible\_host": "serverb", "ansible\_user": "devops"}}, "changed": true}

TASK [echo Hello] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [workstation -> serverb] => {"changed": true, "cmd": ["echo", "Hello from workstation"], "delta": "0:00:00.005530", "end": "2018-04-01 21:15:00.863513", "rc": 0, "start": "2018-04-01 21:15:00.857983", "stderr": "", "stderr\_lines": [], "stdout": "Hello from workstation", "stdout\_lines": ["Hello from workstation"]}

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

ok: [workstation] => {

"msg": "Hello from workstation"

}

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

## Delegated Facts

Any facts gathered by a delegated task are assigned by default to the delegate\_to host, instead of the host which actually produced the facts. The following example shows a task file that will loop through a list of inventory servers to gather facts.

[student@workstation ~]$ **cat inventory**

server1.example.com

server2.example.com

192.168.1.10

192.168.1.11

172.25.250.10

workstation

[lb-servers]

workstation

[web-servers]

serverb

[my-servers2]

servera

serverb

[dc1-webservers]

servera.example.com

serverb.example.com

#192.168.0.[1:254]

#[servers:children]

#my-servers1

#my-servers2

[student@workstation ~]$ **cat delegate-facts.yml**

- name: Delegate Facts

hosts: my-servers2

tasks:

- name: gather facts from app servers

setup:

delegate\_to: "{{item}}"

with\_items: "{{groups['lb-servers']}}"

- name: Debug output

debug:

var: ansible\_enp0s3['ipv4']['address']

[student@workstation ~]$ **ansible-playbook delegate-facts.yml**

PLAY [Delegate Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [gather facts from app servers] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera -> workstation] => (item=workstation)

ok: [serverb -> workstation] => (item=workstation)

TASK [Debug output] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera] => {

"ansible\_enp0s3['ipv4']['address']": "172.25.250.10"

}

ok: [serverb] => {

"ansible\_enp0s3['ipv4']['address']": "172.25.250.10"

}

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

servera : ok=3 changed=0 unreachable=0 failed=0

serverb : ok=3 changed=0 unreachable=0 failed=0

# Configuring Parallelism

## Configure parallelism in Ansible using forks

Ansible allows much more control over the execution of the playbook by running the tasks in parallel on all hosts. By default, Ansible only fork up to five times, so it will run a particular task on five different machines at once. This value is set in the Ansible configuration file ansible.cfg.

[student@demo ~]$ grep forks /etc/ansible/ansible.cfg

#forks = 5

When there are a large number of managed hosts (more than five), the forks parameter can be changed to something more suitable for the environment. The default value can be either overridden in the configuration file by specifying a new value for the forks key, or the value can be changed using the --forks option for the **ansible-playbook** or **ansible** commands.

## Running tasks in parallel

For any specific play, you can use the **serial** keyword in a playbook to temporarily reduce the number of machines running in parallel from the fork count specified in the Ansible configuration file. The serial keyword is primarily used to control rolling updates.

**Rolling updates**

If there is a website being deployed on 100 web servers, only 10 of them should be updated at the same time. The serial key can be set to 10 in the playbook to reduce the number of simultaneous deployments (assuming that the fork key was set to something higher). The serial keyword can also be specified as a percentage which will be applied to the total number of hosts in the play. If the number of hosts does not divide equally into the number of passes, the final pass will contain the modulus. Regardless of the percentage, the number of hosts per pass will always be 1 or greater.

---

- name: Limit the number of hosts this play runs on at the same time

hosts: appservers

serial: 2

[**Rolling Update Batch Size**](http://docs.ansible.com/ansible/latest/playbooks_delegation.html#id8)

New in version 0.7.

By default, Ansible will try to manage all of the machines referenced in a play in parallel. For a rolling updates use case, you can define how many hosts Ansible should manage at a single time by using the **‘’serial’’** keyword:

- name: test play

hosts: webservers

serial: 3

In the above example, if we had 100 hosts, 3 hosts in the group ‘webservers’ would complete the play completely before moving on to the next 3 hosts.

The ‘’serial’’ keyword can also be specified as a percentage in Ansible 1.8 and later, which will be applied to the total number of hosts in a play, in order to determine the number of hosts per pass:

- name: test play

hosts: webservers

serial: "30%"

## Asynchronous tasks

There are some system operations that take a while to complete. For example, when downloading a large file or rebooting a server, such tasks take a long time to complete. Using parallelism and forks, Ansible starts the command quickly on the managed hosts, then polls the hosts for status until they are all finished.

To run an operation in parallel, use the **async** and **poll** keywords. The **async** keyword triggers Ansible to run the job in the background and can be checked later, and its value will be the maximum time that Ansible will wait for the command to complete. The value of **poll** indicates to Ansible how often to **poll** to check if the command has been completed. The default **poll** value is 10 seconds.

In the example, the **get\_url** module takes a long time to download a file and **async: 3600** instructs Ansible to wait for 3600 seconds to complete the task and **poll: 10** is the polling time in seconds to check if the download is complete.

---

- name: Long running task

hosts: demoservers

remote\_user: devops

tasks:

- name: Download big file

get\_url:

url: http://demo.example.com/bigfile.tar.gz

async: 3600

poll: 10

**Deferring asynchronous tasks**

Long running operations or maintenance scripts can be carried out with other tasks, whereas checks for completion can be deferred until later using the wait\_for module. To configure Ansible to not wait for the job to complete, set the value of poll to 0 so that Ansible starts the command and instead of polling for its completion it moves to the next tasks.

[student@workstation ~]$ **cat parallel-async.yml**

---

- name: Restart and wait until the server is rebooted

hosts: my-servers2

serial: 1 -> one server will be shutdown at a time and then next server will be down

remote\_user: devops

tasks:

- name: restart machine

shell: sleep 2 && shutdown -r now "Ansible updates triggered"

async: 1

poll: 0

become: true

ignore\_errors: true

- name: waiting for server to come back

wait\_for:

host: "{{ inventory\_hostname }}"

state: started

delay: 30

timeout: 300

port: 22

delegate\_to: workstation

become: false

[student@workstation ~]$ ansible-playbook parallel-async.yml

PLAY [Restart and wait until the server is rebooted] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [serverb]

ok: [servera]

TASK [restart machine] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [serverb]

changed: [servera]

TASK [waiting for server to come back] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera -> workstation]

ok: [serverb -> workstation]

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

servera : ok=3 changed=1 unreachable=0 failed=0

serverb : ok=3 changed=1 unreachable=0 failed=0

**Asynchronous task status**

While an asynchronous task is running, you can also check its completion status by using ansible async\_status module. The module requires the job or task identifier as its parameter.

[student@workstation ~]$ **cat parallel-async2.yml**

- name: Async status with fire and forget task

hosts: my-servers2

remote\_user: devops

tasks:

- name: Install Mariadb - async task

yum:

name: mariadb

state: installed

async: 1000

poll: 0

register: output

- name: Wait for lagrge file until copy

async\_status:

jid: "{{ output.ansible\_job\_id }}"

register: job\_result

until: job\_result.finished

retries: 30

[student@workstation ~]$ **ansible-playbook parallel-async3.yml -v**

Using /home/student/ansible.cfg as config file

PLAY [Async status with fire and forget tas] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

ok: [servera]

ok: [serverb]

TASK [Install Mariadb - async task] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera] => {"ansible\_job\_id": "209008050268.5103", "changed": true, "finished": 0, "results\_file": "/root/.ansible\_async/209008050268.5103", "started": 1}

changed: [serverb] => {"ansible\_job\_id": "209941501922.5520", "changed": true, "finished": 0, "results\_file": "/root/.ansible\_async/209941501922.5520", "started": 1}

TASK [Wait for MariabDB install until done] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

FAILED - RETRYING: Wait for MariabDB install until done (30 retries left).

changed: [servera] => {"ansible\_job\_id": "209008050268.5103", "attempts": 2, "changed": true, "finished": 1, "msg": "", "rc": 0, "results": ["Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-\n : manager\nThis system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.\nResolving Dependencies\n--> Running transaction check\n---> Package mariadb.x86\_64 1:5.5.47-5.el7 will be installed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n================================================================================\n Package Arch Version Repository Size\n================================================================================\nInstalling:\n mariadb x86\_64 1:5.5.47-5.el7 rhel7repo 9.1 M\n\nTransaction Summary\n================================================================================\nInstall 1 Package\n\nTotal download size: 9.1 M\nInstalled size: 49 M\nDownloading packages:\nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\n Installing : 1:mariadb-5.5.47-5.el7.x86\_64 1/1 \n Verifying : 1:mariadb-5.5.47-5.el7.x86\_64 1/1 \n\nInstalled:\n mariadb.x86\_64 1:5.5.47-5.el7 \n\nComplete!\n"]}

changed: [serverb] => {"ansible\_job\_id": "209941501922.5520", "attempts": 1, "changed": true, "finished": 1, "msg": "", "rc": 0, "results": ["Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-\n : manager\nThis system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.\nResolving Dependencies\n--> Running transaction check\n---> Package mariadb.x86\_64 1:5.5.47-5.el7 will be installed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n================================================================================\n Package Arch Version Repository Size\n================================================================================\nInstalling:\n mariadb x86\_64 1:5.5.47-5.el7 rhel7repo 9.1 M\n\nTransaction Summary\n================================================================================\nInstall 1 Package\n\nTotal download size: 9.1 M\nInstalled size: 49 M\nDownloading packages:\nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\n Installing : 1:mariadb-5.5.47-5.el7.x86\_64 1/1 \n Verifying : 1:mariadb-5.5.47-5.el7.x86\_64 1/1 \n\nInstalled:\n mariadb.x86\_64 1:5.5.47-5.el7 \n\nComplete!\n"]}

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

servera : ok=3 changed=2 unreachable=0 failed=0

serverb : ok=3 changed=2 unreachable=0 failed=0

# Ansible Vault

Ansible may need access to sensitive data such as passwords or API keys in order to configure remote servers. Normally, this information might be stored as plain text in inventory variables or other Ansible files. But in that case, any user with access to the Ansible files or a version control system which stores the Ansible files would have access to this sensitive data. This poses an obvious security risk.

There are two primary ways to store this data more securely:

* Use Ansible Vault, which is included with Ansible and can encrypt and decrypt any structured data file used by Ansible.
* Use a third-party key management service to store the data in the cloud, such as Vault by HashiCorp, Amazon's AWS Key Management Service, or Microsoft Azure Key Vault.

## Create an encrypted file

To create a new encrypted file use the command **ansible-vault create *filename***. The command will prompt for the new vault password and open a file using the default editor. This is **vim**, but may be changed to **vi** in Ansible 2.1. A different editor may be used by setting and exporting the **$EDITOR** variable. For example, to set the default editor to **nano**, **export EDITOR=nano**.

[student@demo ~]$ **ansible-vault create secret.yml**

New Vault password: **redhat**

Confirm New Vault password: **redhat**

Instead of entering the vault password through standard input, a vault password file can be used to store the vault password. This file will need to be carefully protected through file permissions and other means.

[student@demo ~]$ **ansible-vault create --vault-password-file=vault-pass secret.yml**

The cipher used to protect files is AES256 in recent versions of Ansible, but files encrypted with older versions may still use 128-bit AES.

## Edit an existing encrypted file

To edit an existing encrypted file, Ansible Vault provides the command **ansible-vault edit *filename***. This command will decrypt the file to a temporary file and allows you to edit the file. When saved, it copies the content and removes the temporary file.

[student@demo ~]$ **ansible-vault edit secret.yml**

Vault password: **redhat**

**Note**

The **edit** subcommand always rewrites the file, so it should only be used when making changes. This can have implications when the file is kept under version control. The **view** subcommand should always be used to see the file's contents without making changes.

## Change the password for an encrypted file

The vault password can be changed using the command **ansible-vault rekey *filename***. This command can rekey multiple data files at once. It will ask for the original password and the new password.

[student@demo ~]$ **ansible-vault rekey secret.yml**

Vault password: **redhat**

New Vault password: **RedHat**

Confirm New Vault password: **RedHat**

Rekey successful

When a using vault password file, use the **--new-vault-password-file** option:

[student@demo ~]$ **ansible-vault rekey \**

> **--new-vault-password-file=*NEW\_VAULT\_PASSWORD\_FILE* secret.yml**

## Encrypting an existing file

To encrypt a file that already exists, use the command **ansible-vault encrypt *filename***. This command can take the names of multiple files to be encrypted as arguments.

[student@demo ~]$ **ansible-vault encrypt secret1.yml secret2.yml**

New Vault password: **redhat**

Confirm New Vault password: **redhat**

Encryption successful

Use the **--output=OUTPUT\_FILE** option to save the encrypted file with a new name. At most one input file may be used with the **--output** option.

## Viewing an encrypted file

Ansible Vault allows you to view the encrypted file using the command **ansible-vault view *filename***, without opening it for editing.

[student@demo ~]$ **ansible-vault view secret1.yml**

Vault password: **secret**

less 458 (POSIX regular expressions)

Copyright (C) 1984-2012 Mark Nudelman

less comes with NO WARRANTY, to the extent permitted by law.

For information about the terms of redistribution,

see the file named README in the less distribution.

Homepage: http://www.greenwoodsoftware.com/less

my\_secret: "yJJvPqhsiusmmPPZdnjndkdnYNDjdj782meUZcw"

## Decrypting an existing file

An already existing encrypted file can be permanently decrypted by using the command **ansible-vault decrypt *filename***. The **--output** option can be used when decrypting a single file to save the decrypted file under a different name.

[student@demo ~]$ **ansible-vault decrypt secret1.yml --output=secret1-decrypted.yml**

Vault password: **redhat**

Decryption successful

## Encrypting an existing file

Encrypt the existing file **passwd-decrypted.yml** and save the file as **passwd-encrypted.yml**. Use the **ansible-vault encrypt** subcommand with the ***--output***option. Enter **redhat** as the password and confirm by re-entering the password.

1. [student@workstation conf-ansible-vault]$ **ansible-vault encrypt passwd-decrypted.yml --output=passwd-encrypted.yml**

New Vault password: **redhat**

1. Confirm New Vault password: **redhat**

Encryption successful