**Ansible Playbook**

**What are ansible playbooks.**

Ansible playbook are a way to send commands to remote computers in a scripted way. At a basic level, playbooks can be used to manage configuration of and deployments to remote machines. At at more advanced level, they can sequence multi-tier rollouts involving rolling updates, and can delegate actions to other hosts, interacting with monitoring servers and load balancers along the way.

playbooks are designed to be human-readable and are developed in a basic text language, are written in the YAML data serialization format.

**Sample Playbook – to install and configure Apache Server.**

# su – yogesh

# vi /etc/ansible/apache.yml

---

- hosts: appgroup

user: yogesh

become: yes

become\_method: sudo

tasks:

- name: 1. Create Empty file

file: path=/home/yogesh/file\_01 state=touch owner=yogesh group=yogesh mode=0666

- name: 2. Create media directory.

file: path=/home/yogesh/media state=directory owner=yogesh group=yogesh mode=0755 recurse=yes

- name: 3. copy the standard index.html file.

copy: src=/tmp/index.html dest=/var/www/html/index.html mode=0664

- name: 4. copy the data more than "copy Module"

synchronize: src=/home/yogesh/wpdata/ dest=/home/yogesh/ rsync\_opts="--chown=ubuntu:ubuntu"

- name: 5. Install latest version HTTP/Apache RPMs.

yum: name=httpd state=present

- name: 6. Start httpd service.

service: name=httpd state=started enabled=yes

- name: 7. Add apache firwall rule.

command: firewall-cmd --add-port=80/tcp --permanent

- name: 8. Add apache firwall rule.

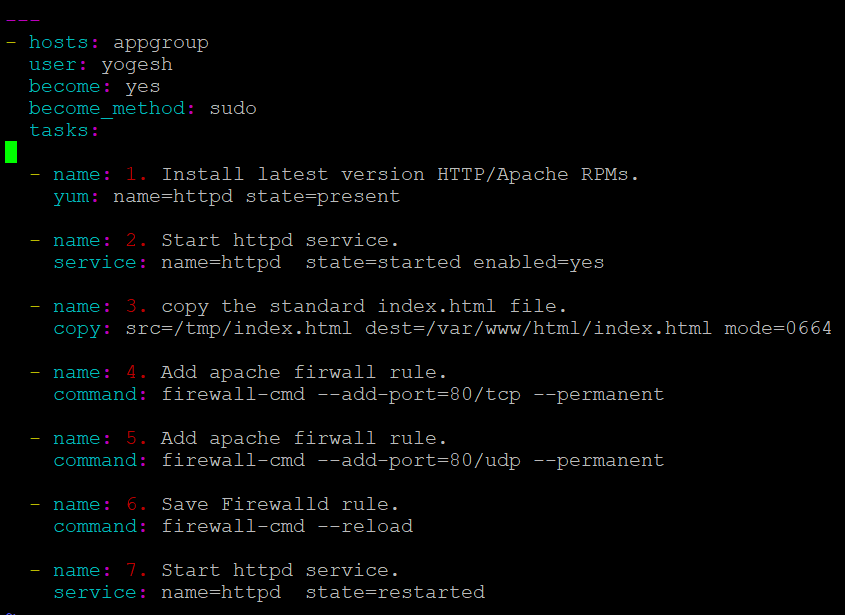
command: firewall-cmd --add-port=80/udp --permanent

- name: 9. Save Firewalld rule.

command: firewall-cmd --reload

- name: 10. Start httpd service.

service: name=httpd state=restarted



**How to Check ansible play book syntax is correct or not.**

# ansible-playbook apache.yml --check

# ansible-playbook apache.yml --syntax-check

**How to run this playbook in remote node.**

# ansible-playbook apache.yml -i /etc/ansible/hosts -l appgroup

# ansible-playbook apache.yml -i /etc/ansible/hosts -l dbgroup

# ansible-playbook apache.yml -i /etc/ansible/hosts -l all

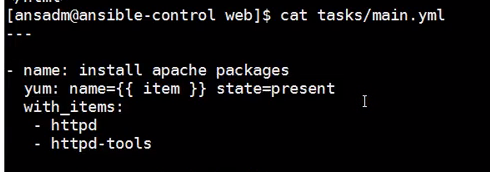
i => Inventory file

-l => to mention group name.

**How to know how many servers impacted with this playbook.**

# ansible-playbook apache.yml --list-hosts

**How to install multiple services in playbook at a time.**



**Playbook for Multiple OS like Centos 6, Centos7 and Ubuntu.**

# vi centos\_ubuntu.yaml

---

- hosts: all

user: yogesh

become: yes

become\_method: sudo

tasks:

- debug:

var: ansible\_distribution

- debug:

var: ansible\_distribution\_major\_version

- debug:

var: ansible\_os\_family

- name: install system updates for centos systems

yum: name=\* state=latest update\_cache=yes

when: ansible\_distribution == "CentOS"

- name: install system updates for ubuntu systems

apt: upgrade=dist update\_cache=yes

when: ansible\_distribution == "Ubuntu"

- name: 1. Run the equivalent of "apt-get update" as a separate step

apt:

update\_cache: yes

when: ansible\_distribution == "Ubuntu"

- name: 2. Run the equivalent of "apt-get upgrade" as a separate step

apt:

name: apt

state: latest

when: ansible\_distribution == "Ubuntu"

- name: 1. Install latest version HTTP/Apache RPMs.

yum: name=httpd state=present

when: ansible\_distribution == "CentOS"

- name: 2. Start httpd service.

service: name=httpd state=started enabled=yes

when: ansible\_distribution == "CentOS"

- name: 1. Install latest version HTTP/Apache RPMs.

yum: name=apache state=present

- name: 2. Start httpd service.

service: name=apache state=started enabled=yes

when: ansible\_distribution == "Ubuntu"

- name: selinux permissive

selinux: policy=targeted state=permissive

when: ansible\_distribution == "CentOS"

- name: firewalld stopped

service: name=firewalld state=stopped enabled=no

**Exmaple:**

- name: "shut down CentOS 6 and Debian 7 systems"

command: /sbin/shutdown -t now

when: (ansible\_distribution == "CentOS" and ansible\_distribution\_major\_version == "6") or

(ansible\_distribution == "Debian" and ansible\_distribution\_major\_version == "7")

- name: "shut down CentOS 6 systems"

command: /sbin/shutdown -t now

when:

- ansible\_distribution == "CentOS"

- ansible\_distribution\_major\_version == "6"

- include: centos5-setup.yml

when: ansible\_distribution == "CentOS" and ansible\_distribution\_major\_version < '6'

- include: centos6-setup.yml

when: ansible\_distribution == "CentOS" and ansible\_distribution\_major\_version == '6'

- include: centos7-setup.yml

when: ansible\_distribution == "CentOS" and ansible\_distribution\_major\_version > '6'

- include: ubuntu-setup.yml

when: ansible\_distribution == "Ubuntu"

OR

# Include OS specific tasks

- include: CentOS.yml

when: ansible\_distribution == "CentOS"

- include: Ubuntu.yml

when: ansible\_distribution == "Ubuntu"

how to stop playbook while running without error.

# do not run on CentOS/RHEL 6

- meta: end\_play

when: ansible\_distribution\_major\_version == '7'

**User Management**

User management is one of the day 2 day system Administration tasks for the system Administrators.

Though this is not a difficult task, but sometime it becomes time consuming in the following scenarios.

* you dont have control user management tool in your setup, such as microsoft AD or LDAP.
* Have to create local users for system/Application users, which can't be AD integrated.

For example, if someone joins your company and you have to create user account on all your servers, it becomes a headache and always a chance of error in the process.

Aforementioned statement also true when someone leaves the company and you have to get rid of the orphan user account.

**How if we can automate this process?**

Yes, that's true, we can. With ansible you just need to update the playbook with the desired user and related information and ansible can create this task for you in seconds on multiple servers.

I have written a small playbook, which can be used for

* Create local groups
* create loal users
* Generate ssh keys for a user.
* Update the public keys in user's authorized\_key file.
* Delete local groups.
* Delete local users.

**User delete manually**

# userdel -r demouser1

**1)Playbook for users management**

# vi usermanagement.yaml

---

## This playbook use "User, group, authorized\_keys" modules.

- hosts: localhost

become\_user: root

become: yes

tasks:

# This task create Groups

- name: add a group

group:

name={{ item }}

state=present

with\_items:

- demogrp

- demogrp1

- demogrp2

tags: add\_new\_group

# This task adds several users on the target machines for /standard loop used

- name: add serveral users

user:

name={{ item }}

state=present

groups=demogrp

password="$1$tata$3qsaTRC.H2dIu7JpS3gTM0"

shell=/bin/bash

with\_items:

- demouser1

- demouser2

- demouser3

- demouser4

tags: add\_new\_user

# This task generate public keys for user.

- name: generate ssh keys for a user

user:

name={{ item }}

generate\_ssh\_key=yes

ssh\_key\_bits=4096

ssh\_key\_file=.ssh/id\_rsa

with\_items:

- demouser1

- demouser2

- demouser3

tags: generate\_ssh\_keys

# This task is to copy the public keys to user account.

- name: update authorized key for a user.

authorized\_key:

user: demouser1

state: present

key: "{{ lookup('file', '/home/demouser1/.ssh/id\_rsa.pub') }}"

tags: copy\_pub\_key

# This tasks is to delete the users.

- name: delete several users

user:

name={{ item }}

state=absent

with\_items:

- demouser3

- demouser4

tags: remove\_user

# This tasks is to delete the group

- name: delete several group

group:

name={{ item }}

state=absent

with\_items:

- demogrp1

- demogrp2

tags: remove\_group

**2)User creation with playbook.**

# vi usercreation.yaml

---

- hosts: all

user: yogesh

become: yes

become\_method: sudo

vars:

user\_pass: $6$rMKAz.bu5XsnFxJ3$b2bgRSw0hHPYEcZEjCtFiIYgzAG/ZzcIAflg6xTcmaDdxyOSjrrMAHERCbcAy6yNpaarIwGKvjDBcsPnVcZmr/

root\_pass: $6$rMKAz.bu5XsnFxJ3$b2bgRSw0hHPYEcZEjCtFiIYgzAG/ZzcIAflg6xTcmaDdxyOSjrrMAHERCbcAy6yNpaarIwGKvjDBcsPnVcZmr/

tasks:

- name: create new user

user:

name: amit

uid: 1111

comment: yogesh Bhagwatkar

password: "{{ user\_pass }}"

shell: /bin/bash

tags: create\_user

- name: root password change

user: name=root update\_password=always password="{{ root\_pass }}"

tags: update\_root\_pass

**3)How to check how many tags we have in playbooks.**

# ansible-playbook usermanagement.yaml --list-tags

**4)How to run only one tag of the recipe.**

# ansible-playbook usermanagement.yaml --tags <Tag Name>

# ansible-playbook usermanagement.yaml --tags add\_new\_group

# ansible-playbook usermanagement.yaml --tags add\_new\_user

**5)Run the playbook tag vise.**

# ansible-playbook usermanagement.yaml --tags generate\_ssh\_key

# ansible-playbook usermanagement.yaml --tags copy\_pub\_key

# ansible-playbook usermanagement.yaml --tags remove\_group

# ansible-playbook usermanagement.yaml --tags remove\_user

Note : All tag name will be present in usermanagement.yaml file. We are running playbook tag vies.

**How to login in different system without password (Key based authentication)**

# ssh demouser1@192.168.1.11 -i /home/demouser1/.ssh/id\_rsa

**Configuring and managing the NTP**

Configuring and managing the NTP client on the enterprise servers is one of the tasks for system administrators.

NTP client is responsible for the time sync of client server with the time sources. Assume a scenario, you have to configure NTP Client on multiple servers and in manual way it is going to take long time and always chance of error.

How if we Ansible to perform this tasks?

I have written a small playbook, which you can customize further to install, configure and manage the NTP client on Linux/Unix systems. Below modules as part of tasks

1)Yum

2)Template

3)Service

Along with modules, I have used variables and handlers to make this demo playbook more efficient.

**How to check ntp status of all servers.**

# ansible all -i /etc/ansible/hosts -m shell -a "date"

# ansible all -i /etc/ansible/hosts -m shell -a "ntpq -p"

# i => Inventory

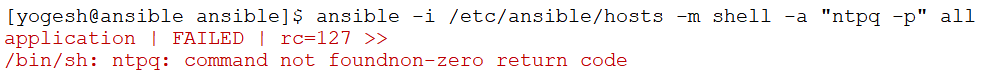
# -m => Module

# -a => Arguments

**How to check NTP Client status**

# ansible all -i /etc/ansible/hosts -m shell -a "ntpq -p"

Output:



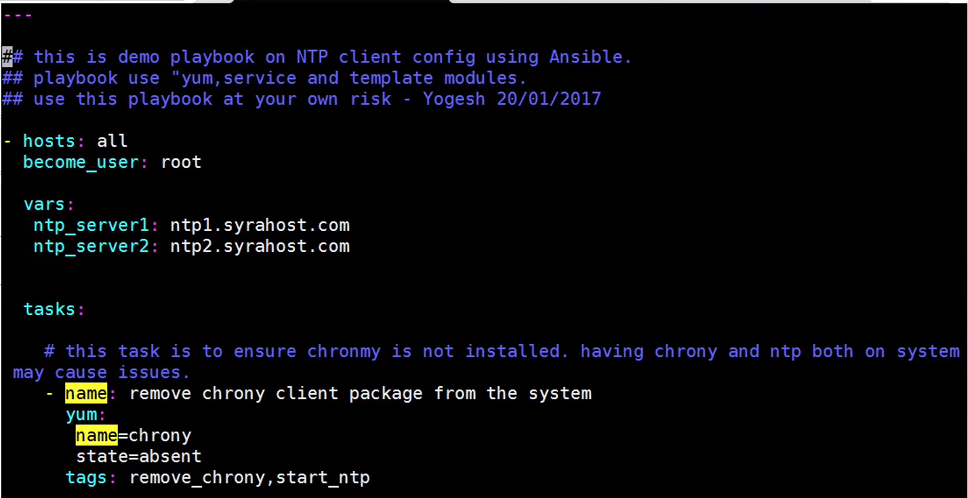
Note : Mins NTP package not install in this system.

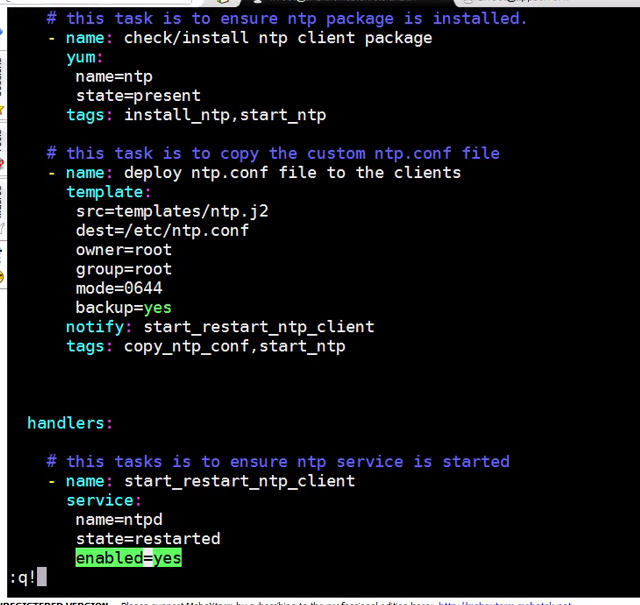
**How to check NTP package installed or not in all servers.**

# ansible all -i /etc/ansible/hosts -m shell -a "rpm -qa | grep ntp"

**Playbook for installation of NTP client.**

# vi ntpmanagement.yaml





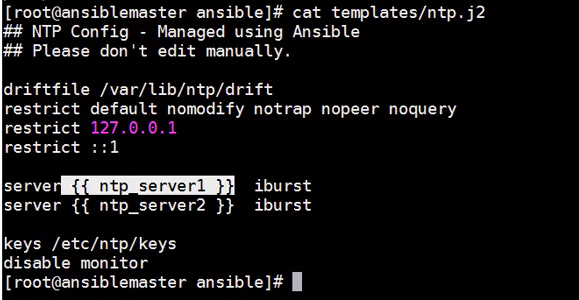
**How to run ansible playbook without tags.**

# ansible-playbook ntpmanagement.yaml

**Template file are**

# cd /etc/ansible/templates/

# vi ntp.j2

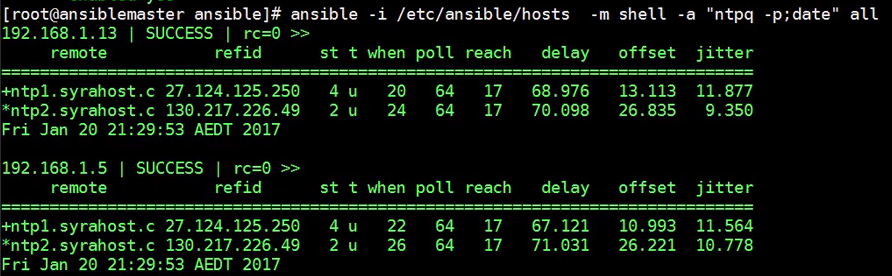


ntp.j2

# .j2 => Jinja 2 formate.

**How to check date on target system.**

# ansible all -i /etc/ansible/hosts -m shell -a "rpm -qa | grep ntp;date"



**Password Hashing in Ansible.**

**What is password Hashing?**

Password hashing is method to take variable length password as input and creating a cryptic, fixed length password from it using the cryptic mechanisms.

To make your hashed password more secure, you can add salt to your input. Salt is generally a random value used to generate hashed password.

**Benefits of using the hashing.**

Instead of storing the plain password of user, storing hashed password makes more security.

Hashing is one-way function, revering the hashed password to normal string is very difficult.

Hashed password can be used for managing users in automation tools etc.

**Methods to Generate Hashed Passwords.**

There are different hashing algorithms, the most commonly used being MD5 and SHA. Sha-512 is the secure algorithms on date

**How to check default linux password hashing algorithm we are using.**

# authconfig --test | grep “password hashing algorithm”

Output:

password hashing algorithm is sha512

**How to check remote server password hashing algorithm**

# ansible <Remote Server Name> -m shell -a "authconfig --test | grep 'password hashing algorithm'" -s

# ansible application -m shell -a "authconfig --test | grep 'password hashing algorithm'" -s

Output:

application | SUCCESS | rc=0 >>

password hashing algorithm is sha512

**How to Generate Encrypted password**

**1)From python How to create Random Hashed/ Encrypted password (From any user for any user).**

# python -c 'import crypt,getpass; print crypt.crypt(getpass.getpass())'

Output:

$6$fHQbeVFEKdSMh50w$tDiw8rNvh0S1bqrCZTRTugRQkzAZzo9DVi6WeP80ALjDNuFZuAfFFLGiB0J.41LobTrXPzFlUyhuhThpkO/bc.

Note: getpass()) = we need to put any password which we want to encrypt.

**2) From Open SSL How to create Random Hashed/ Encrypted password.**

# openssl passwd -1 -salt <Salt Value> <Password>

# openssl passwd -1 -salt $(openssl rand -base64 6) Test123

Output:

$1$EirWs0Br$9bAR2Nk2v3YAkoqE4H4Je.

Note: $(openssl rand -base64 6) = it’s a random slat value for password.

**Crete playbook for hashpass word and mention this encrypted password inside the playbook.**

# vim hashpass.yml

---

- hosts: all

user: yogesh

become: yes

become\_method: sudo

vars:

user\_pass: $6$rMKAz.bu5XsnFxJ3$b2bgRSw0hHPYEcZEjCtFiIYgzAG/ZzcIAflg6xTcmaDdxyOSjrrMAHERCbcAy6yNpaarIwGKvjDBcsPnVcZmr/

root\_pass: $6$rMKAz.bu5XsnFxJ3$b2bgRSw0hHPYEcZEjCtFiIYgzAG/ZzcIAflg6xTcmaDdxyOSjrrMAHERCbcAy6yNpaarIwGKvjDBcsPnVcZmr/

tasks:

- name: create new user

user:

name: amit

uid: 1111

comment: yogesh Bhagwatkar

password: "{{ user\_pass }}"

shell: /bin/bash

tags: create\_user

- name: root password change

user: name=root update\_password=always password="{{ root\_pass }}"

tags: update\_root\_pass

**How to ping all the server.**

# ansible all -m ping

**Check how many tags are available in on playbook.**

# ansible-playbook hashpass.yml --list-tags

Output:

playbook: hashpass.yml

play #1 (all): all TAGS: []

TASK TAGS: [create\_user, update\_root\_pass]

**How to run Create\_user task tag.**

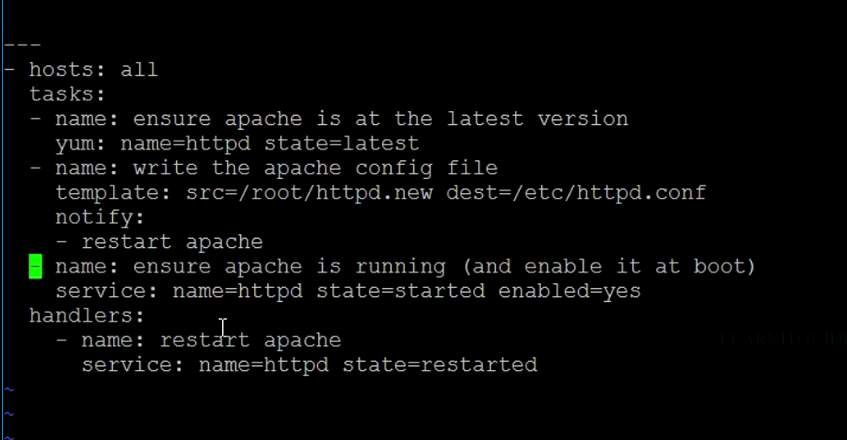
# ansible-playbook hashpass.yml -t create\_user

# ansible-playbook hashpass.yml

**How to check created password in remote system.**

# chage -l yogesh

**Ansible playbook for apache**



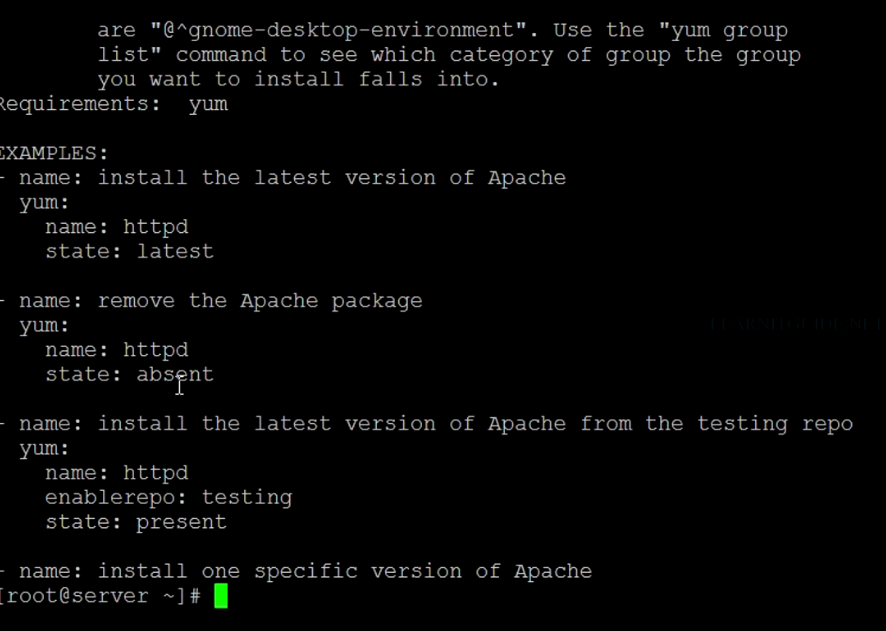
# if you want to get help for modules or state.

# ansible-doc -l | more

For module

# ansible-doc <Module name>

# ansible-doc yum



**How many host effected from playbook.**

# ansible-playbook apache.yaml --llist-host

**==================================================================================================**

**Set Hostname to multiple remote hosts using ansible**

- This playbook will set hostname to multiple remote hosts using ansible.

- It will get hostname from inventory file ( /etc/ansible/hosts ) and setup remote hosts as per that system IP Add.

**1) Create Host file in /etc/ansible/hosts**

**# vim /etc/ansible/hosts**

[all]

test-01 ansible\_ssh\_host=10.21.101.32 ansible\_user=ubuntu ansible\_python\_interpreter=/usr/bin/python3

test-02 ansible\_ssh\_host=10.21.101.33 ansible\_user=ubuntu ansible\_python\_interpreter=/usr/bin/python3

test-03 ansible\_ssh\_host=10.21.101.34 ansible\_user=ubuntu ansible\_python\_interpreter=/usr/bin/python3

test-04 ansible\_ssh\_host=10.21.101.35 ansible\_user=ubuntu ansible\_python\_interpreter=/usr/bin/python3

**2) Create ansible playbook for change hostname.**

**# vim /etc/ansible/change-hostname.yaml**

---

- hosts: all

user: ubuntu

become: yes

become\_method: sudo

tasks:

- name: Change the hostname to our standard

hostname:

name : "{{ inventory\_hostname }}"

when:

ansible\_fqdn != ansible\_ssh\_host

- name: Fix /etc/hosts removing the old hostname

lineinfile:

state: present

dest: /etc/hosts

line: "{{ ansible\_default\_ipv4.address }} {{ inventory\_hostname }} {{ ansible\_hostname }}"

regexp: "^{{ ansible\_default\_ipv4.address }}"

when:

ansible\_fqdn != inventory\_hostname

### Change Single system hostname with localhost entry in /etc/hosts

# - name: change hostname to myserver

# hostname:

# name: "webserver"

#

# - name: add myself to /etc/hosts

# lineinfile:

# dest: /etc/hosts

# regexp: '^127\.0\.0\.1[ \t]+localhost'

# line: '127.0.0.1 localhost webserver'

# state: present

**URL :** <https://github.com/codylane/ansible-playbook-change-hostname/blob/master/change_hostname.yaml>

**URL :** <https://github.com/ansible/ansible-examples/issues/254>