

# Ansible Handbook

---

V 1.0

Tariq Mehmood  
SHERDIL IT ACADEMY

## Contents

Ansible .....	2
Key terms of Ansible: .....	2
Setting up Ansible .....	3
1. Configuring Ansible Controller .....	3
2. Configuring Ansible hosts .....	8
3. Setting up SSH .....	9
Ansible ad-hoc Commands .....	11
Ansible Modules .....	12
Ansible Playbook .....	14
Key Elements of a Playbook: .....	14
Extra features of a Playbook: .....	17
1. Condition (when): .....	17
2. Loops: .....	17
3. Tags: .....	18
Ansible Vault: .....	19
How to Use Ansible Vault: .....	19

# Ansible

Ansible is an open-source automation tool used to automate tasks in IT systems. It works as a remote controller for managing computers, servers, and other devices. Ansible can be used to setting up software, configuring systems, or running commands on each device.

It works by using simple ad-hoc commands or by using book of instructions called playbooks written in a language called YAML. These playbooks tell Ansible what actions to perform on host devices.

It connects to target systems via SSH or WinRM. It's agentless, meaning no special software needs to be installed on the managed nodes.

It is push-based this means that the control node pushes out configurations or tasks to the host nodes when you execute a ad-hoc command /playbook.

It is known for its scalability, ease of use, and ability to manage diverse environments, including servers, cloud platforms, and network devices, all while ensuring consistency and reliability across systems.

## Key terms of Ansible:

- **Ansible Controller:** The machine from which Ansible is run.
- **Module:** A unit of work Ansible uses to perform tasks.
- **Task:** A single action within a playbook.
- **Role:** A reusable set of tasks and configurations.
- **Fact:** Information about a system gathered by Ansible.
- **Play:** A section of a playbook targeting specific hosts with specific tasks.
- **Host:** A target machine managed by Ansible.
- **Inventory:** A file listing all hosts and groups of hosts for automation

# Setting up Ansible

It can be divided in 3 parts:

1. Configuring Ansible Controller
2. Configuring Ansible hosts
3. Setting up SSH

## 1. Configuring Ansible Controller

1<sup>st</sup> we create ansible user and provide it sudo privileges with following commands (for debian based system):

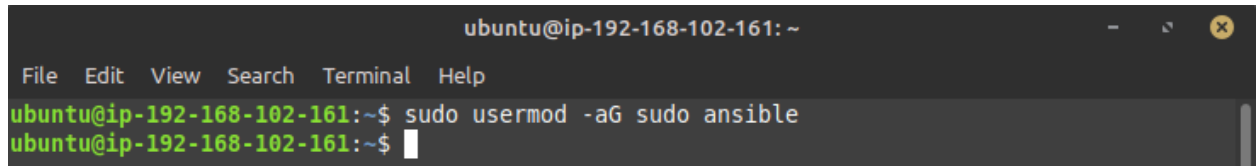
Creating user ansible:

>> sudo adduser ansible

```
ubuntu@ip-192-168-102-161:~$ sudo adduser ansible
Adding user `ansible' ...
Adding new group `ansible' (1006) ...
Adding new user `ansible' (1005) with group `ansible' ...
Creating home directory `/home/ansible' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ansible
Enter the new value, or press ENTER for the default
  Full Name []: ansible
    Room Number []:
    Work Phone []:
    Home Phone []:
      Other []:
Is the information correct? [Y/n] y
ubuntu@ip-192-168-102-161:~$
```

Giving ansible user sudo privileges:

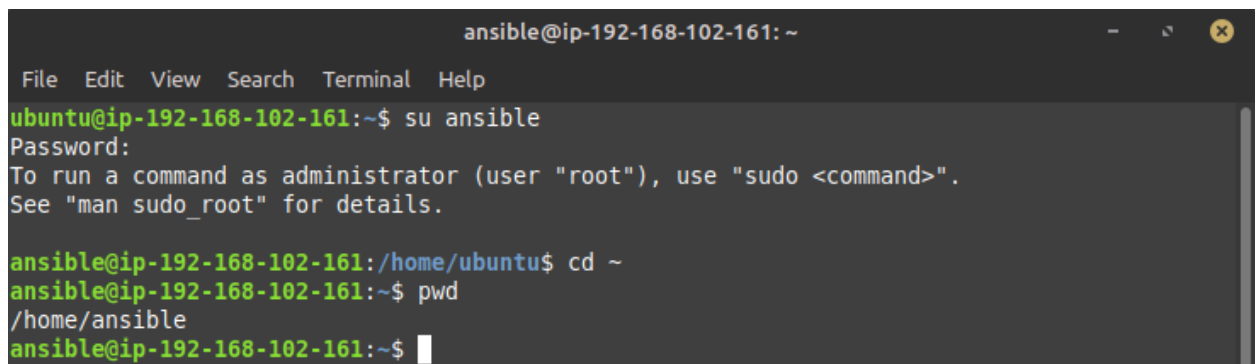
>> sudo usermod -aG sudo ansible

A terminal window titled 'ubuntu@ip-192-168-102-161: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'ubuntu@ip-192-168-102-161:~\$' and the command 'sudo usermod -aG sudo ansible' has been entered. The cursor is at the end of the command line.

```
ubuntu@ip-192-168-102-161:~$ sudo usermod -aG sudo ansible
ubuntu@ip-192-168-102-161:~$
```

Switching to ansible user:

>> su ansible

A terminal window titled 'ansible@ip-192-168-102-161: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'ubuntu@ip-192-168-102-161:~\$' and the command 'su ansible' has been entered. It prompts for a password, then shows instructions for using sudo. The prompt changes to 'ansible@ip-192-168-102-161:/home/ubuntu\$' and the command 'cd ~' is entered. The prompt changes to 'ansible@ip-192-168-102-161:~\$' and the command 'pwd' is entered, showing the output '/home/ansible'.

```
ansible@ip-192-168-102-161:~$ su ansible
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ansible@ip-192-168-102-161:/home/ubuntu$ cd ~
ansible@ip-192-168-102-161:~$ pwd
/home/ansible
ansible@ip-192-168-102-161:~$
```

Now we have created ansible user on controller now we need to install and configure ansible on controller with following commands:

Installing ansible:

>> sudo apt install software-properties-common

>> sudo add-apt-repository --yes --update ppa:ansible/ansible

>> sudo apt update

>> sudo apt install ansible

```
ansible@ip-192-168-102-161: ~
File Edit View Search Terminal Help
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:5 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
22 packages can be upgraded. Run 'apt list --upgradable' to see them.
ansible@ip-192-168-102-161:~$ sudo apt install ansible
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  apache2-bin apache2-data apache2-utils libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  ansible-core python3-jmespath python3-kerberos python3-nacl python3-ntlm-auth python3-paramiko
  python3-requests-kerberos python3-requests-ntlm python3-resolvelib python3-winrm python3-xlrd python3-xlsxwriter
  sshpass
Suggested packages:
  python-nacl-doc python3-gssapi python3-invoke
The following NEW packages will be installed:
  ansible ansible-core python3-jmespath python3-kerberos python3-nacl python3-ntlm-auth python3-paramiko
  python3-requests-kerberos python3-requests-ntlm python3-resolvelib python3-winrm python3-xlrd python3-xlsxwriter
  sshpass
0 upgraded, 13 newly installed, 0 to remove and 22 not upgraded.
Need to get 18.9 MB of archives.
After this operation, 208 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

## Verifying ansible installation

>> ansible --version

```
ansible@ip-192-168-102-161: ~
File Edit View Search Terminal Help
ansible@ip-192-168-102-161:~$ ansible --version
ansible [core 2.17.6]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ansible/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/ansible/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.10.12 (main, Sep 11 2024, 15:47:36) [GCC 11.4.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
ansible@ip-192-168-102-161:~$
```

## Basic Configuration:

>> cd /etc/ansible

>> sudo su

>> ansible-config init -t all --disabled > ansible.cfg

>> exit

```
ansible@ip-192-168-102-161: /etc/ansible
File Edit View Search Terminal Help
ansible@ip-192-168-102-161:~$ cd /etc/ansible
ansible@ip-192-168-102-161:/etc/ansible$ ls
ansible.cfg  hosts  roles
ansible@ip-192-168-102-161:/etc/ansible$ sudo su
root@ip-192-168-102-161:/etc/ansible# ansible-config init -t all --disabled > ansible.cfg
root@ip-192-168-102-161:/etc/ansible# exit
exit
ansible@ip-192-168-102-161:/etc/ansible$
```

>> sudo nano /etc/ansible/ansible.cfg

#Added the following configuration settings after the [defaults]:

>> remote\_user=ansible

>> host\_key\_checking=False

```
ansible@ip-192-168-102-161: /etc/ansible
File Edit View Search Terminal Help
GNU nano 6.2 /etc/ansible/ansible.cfg *
# (list) List of valid system temporary directories on the managed machine for Ansible to validate O(remote_tmp) against,
# When O(remote tmp) is required to be a system temp dir and it does not match any in the list, the first one from the list
;system_tmpdirs=/var/tmp, /tmp

# (boolean) This makes the temporary files created on the machine world-readable and will issue a warning instead of failing
# It is useful when becoming an unprivileged user.
;allow_world_readable_tmpfiles=False

inventory= ~/ansible/config/hosts
remote_user=ansible
host_key_checking=False

[privilege_escalation]
# (boolean) Display an agnostic become prompt instead of displaying a prompt containing the command line supplied become module
;agnostic_become_prompt=True

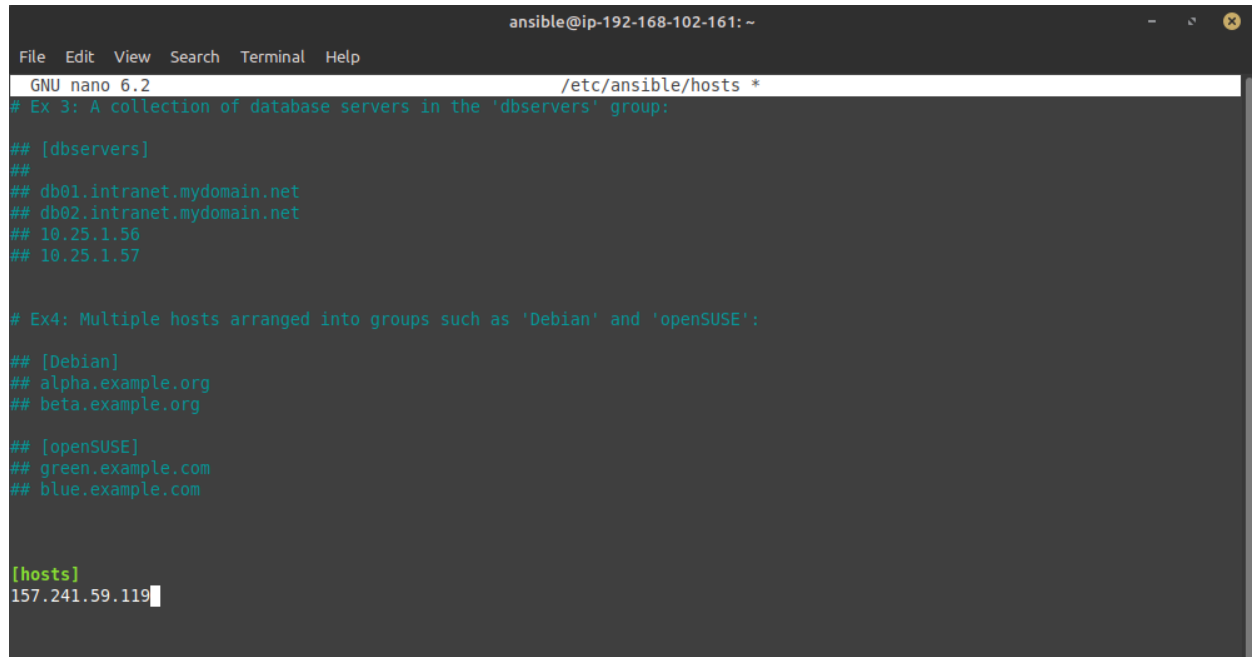
# (boolean) When ``False`` (default), Ansible will skip using become if the remote user is the same as the become user, as
# If ``True``, this forces Ansible to use the become plugin anyways as there are cases in which this is needed.
;become_allow_same_user=False

# (boolean) Toggles the use of privilege escalation, allowing you to 'become' another user after login.
;become=False

# (boolean) Toggle to prompt for privilege escalation password.
File Name to Write: /etc/ansible/ansible.cfg
^G Help M-D DOS Format M-A Append M-B Backup File
^C Cancel M-M Mac Format M-P Prepend ^T Browse
```

Adding host group & host IPs in inventory

>> sudo nano /etc/ansible/hosts

A screenshot of a terminal window with a dark background. The title bar at the top reads 'ansible@ip-192-168-102-161: ~'. Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main area shows the 'GNU nano 6.2' editor editing the file '/etc/ansible/hosts \*'. The file content includes a comment about database servers, a group definition for 'dbservers' with four members, a comment about Debian and openSUSE, and two more group definitions for 'Debian' and 'openSUSE' with three members each. At the bottom, there is a group definition for 'hosts' with one member, '157.241.59.119', which is currently being edited, indicated by a cursor and a blinking line.

```
ansible@ip-192-168-102-161: ~
File Edit View Search Terminal Help
GNU nano 6.2 /etc/ansible/hosts *
# Ex 3: A collection of database servers in the 'dbservers' group:

## [dbservers]
##
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57

# Ex4: Multiple hosts arranged into groups such as 'Debian' and 'openSUSE':

## [Debian]
## alpha.example.org
## beta.example.org

## [openSUSE]
## green.example.com
## blue.example.com

[hosts]
157.241.59.119
```



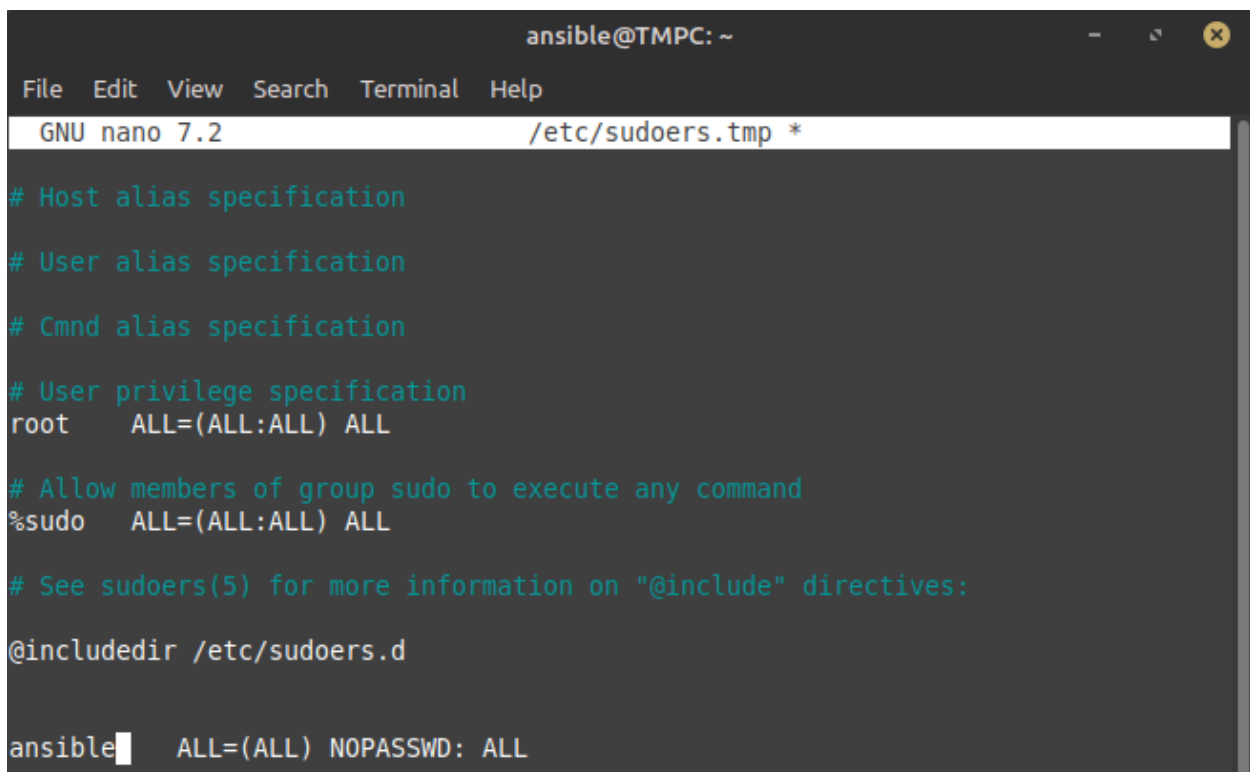
## 2. Configuring Ansible hosts

Creating ansible user with sudo privileges and installing ansible in host systems with following commands (for debian based system):

```
>> sudo adduser ansible
>> sudo usermod -aG sudo ansible
>> su ansible
>> sudo apt install software-properties-common
>> sudo add-apt-repository --yes --update ppa:ansible/ansible
>> sudo apt update
>> sudo apt install ansible
```

Setting up password-less sudo privileges on host

```
>> sudo visudo
# Adding following in file
>> ansible ALL=(ALL) NOPASSWD: ALL
```



```
ansible@TMPC: ~
File Edit View Search Terminal Help
GNU nano 7.2 /etc/sudoers.tmp *

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:

@includedir /etc/sudoers.d

ansible  ALL=(ALL) NOPASSWD: ALL
```

### 3. Setting up SSH

Creating ssh-keygen for ansible user on ansible controller:

```
>> ssh-keygen -t rsa
```

```
ansible@ip-192-168-102-161: ~  
File Edit View Search Terminal Help  
ansible@ip-192-168-102-161:~$ ssh-keygen -t rsa  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/ansible/.ssh/id_rsa):  
Created directory '/home/ansible/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/ansible/.ssh/id_rsa  
Your public key has been saved in /home/ansible/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:eabZYxKBUB0C85PjEATcSQDtmm7GnTleL8pPC8meRQ0 ansible@ip-192-168-102-161  
The key's randomart image is:  
+---[RSA 3072]-----+  
|.0=0...0  
| 0.0. 0 0  
|. +E0 0  
|. +00 0  
|0 ..=S 0  
|0 . 00 0 B  
|0 .+0+. + +  
|=0=*0 0 .  
|0 .*+0..  
+---[SHA256]-----+  
ansible@ip-192-168-102-161:~$
```

Viewing Ansible controller public ssh key

```
>>cat .ssh/id_rsa.pub
```

```
ansible@ip-192-168-101-208: ~  
File Edit View Search Terminal Help  
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQgQDLwhWAeRS095K+5+7ky3+fgThYuQmS/q7R3SVjghHB  
prnuZYrkblNoFiQCY2sj9uYSYpU2RDdF9k3bUR3cjKSSMAXCdkYIJHPRareWJi5+zvh8jzgJXeUHarrI  
5iKEk9LCkoHmCpN77cy2Q0NvKqYrW5oGVX1/6TcwzLAqdQusGRrwfgsfESwQWwHTJFPN2KaVj/3g4tb  
a2Y2XjGh3lhbXmg8H002LqK5YTJj1jYHAH/BxVmjiUZYolw5+Ied0w0egzP5L0aNjlv446kEb3evtJe  
tgwYXbCSDRNA0C0BKhacPzDLmtYCLrDs4syh7pjRwbIXjd0yGjSmCmtQvcuvb0GS7205k0GQr583aP17  
+uWGQcPltgyNzsuDIqjTHqsk22Z8iHwavjawT4mXH/AJhJX1lV4+PiYqUlHbbD8MrPJTbM2LEQeuuRjB  
vVvAsBrjd7tNIZjXQM5UUPycY6BGicraWVW3UMr9LiQTQplKTHrs4+Vk0xcgJ+D4cpsgxa0= ansible  
@ip-192-168-102-161
```

Sharing ssh with following command:

>>ssh-copy-id ansible@host\_ip

```
ansible@TMPC: ~  
File Edit View Search Terminal Help  
ansible@TMPC:~$ ssh-copy-id ansible@192.168.0.121  
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ansible/.ssh/id_rsa.pub"  
The authenticity of host '192.168.0.121 (192.168.0.121)' can't be established.  
ED25519 key fingerprint is SHA256:b4tyQsAzSD2+mAICSPmexvwWy1V6l0bwxauNKuUj5mM.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed  
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys  
ansible@192.168.0.121's password:  
  
Number of key(s) added: 1  
  
Now try logging into the machine, with: "ssh 'ansible@192.168.0.121'"  
and check to make sure that only the key(s) you wanted were added.  
  
ansible@TMPC:~$
```

Verifying ssh sharing

>>ssh ansible@host\_ip

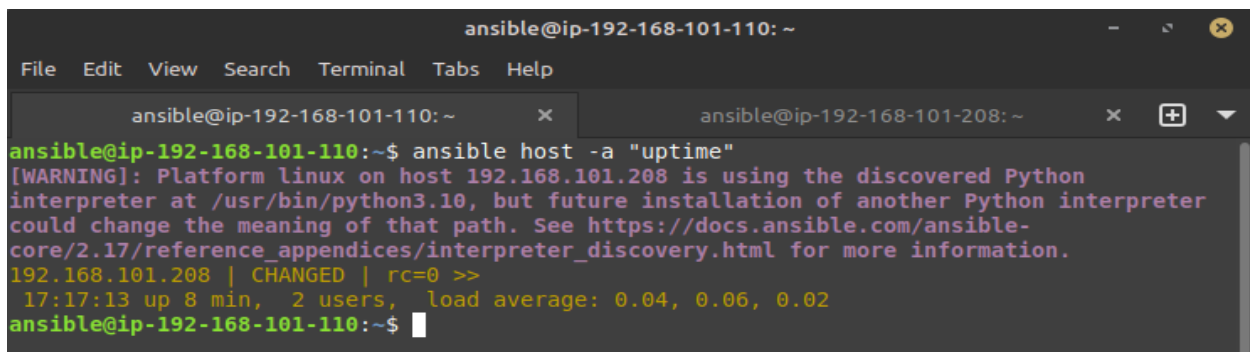
```
ansible@agent01: ~  
File Edit View Search Terminal Help  
ansible@TMPC:~$ ssh ansible@192.168.0.121  
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-125-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
New release '22.04.5 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Tue Dec 3 21:04:02 2024 from 192.168.0.120  
ansible@agent01:~$
```

# Ansible ad-hoc Commands

In Ansible, an ad-hoc command is a one-time command that allows you to execute tasks on host systems without creating a full playbook. This is useful for quick, simple tasks. We can execute ad-hoc commands directly (without modules) with `-a` flag some example for those commands are followings:

>> `ansible <hosts> -a "<shell command>"`

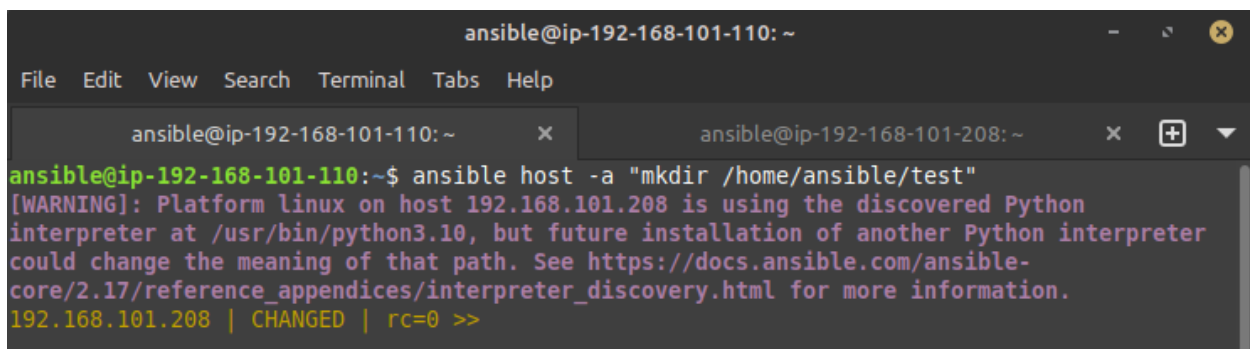
i. `ansible host -a "uptime"`



```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help

ansible@ip-192-168-101-110: ~ x ansible@ip-192-168-101-208: ~ x + v
ansible@ip-192-168-101-110:~$ ansible host -a "uptime"
[WARNING]: Platform linux on host 192.168.101.208 is using the discovered Python
interpreter at /usr/bin/python3.10, but future installation of another Python interpreter
could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more information.
192.168.101.208 | CHANGED | rc=0 >>
17:17:13 up 8 min, 2 users, load average: 0.04, 0.06, 0.02
ansible@ip-192-168-101-110:~$
```

ii. `ansible host -a "mkdir /home/ansible/test"`



```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help

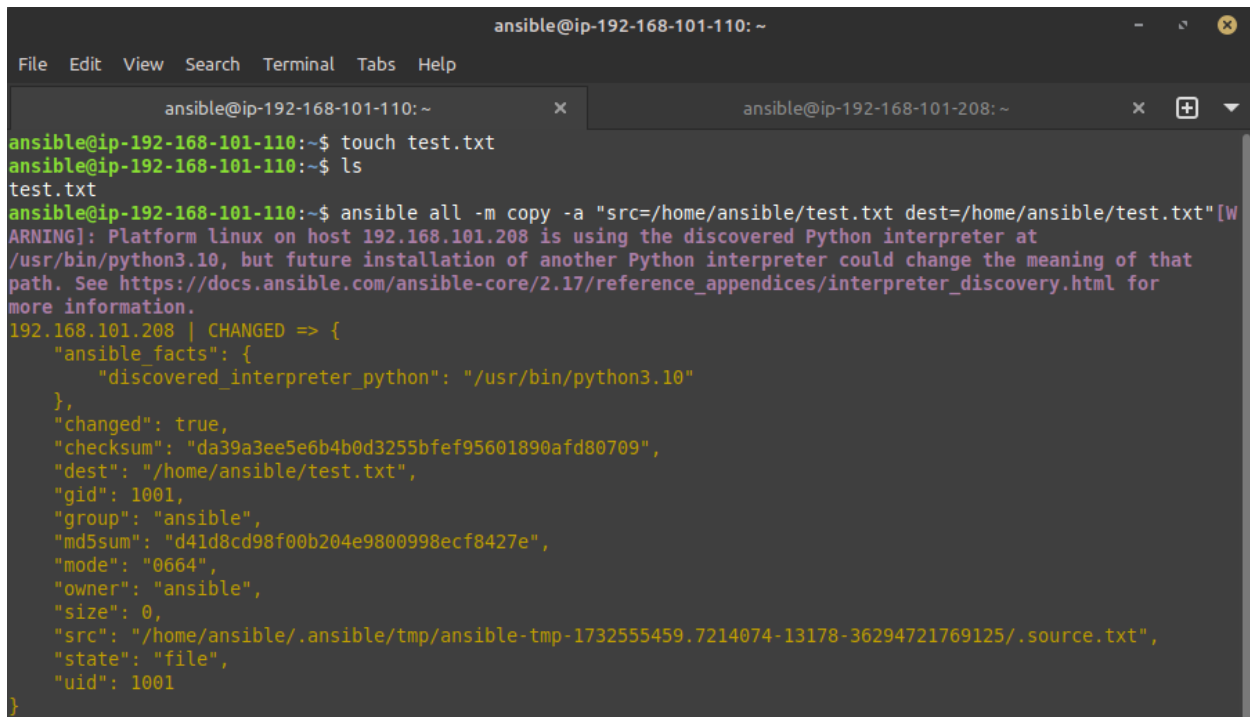
ansible@ip-192-168-101-110: ~ x ansible@ip-192-168-101-208: ~ x + v
ansible@ip-192-168-101-110:~$ ansible host -a "mkdir /home/ansible/test"
[WARNING]: Platform linux on host 192.168.101.208 is using the discovered Python
interpreter at /usr/bin/python3.10, but future installation of another Python interpreter
could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more information.
192.168.101.208 | CHANGED | rc=0 >>
```

# Ansible Modules

We can also use predefined modules to perform task easily some example are followings:

>> ansible <hosts> -m <module> -a "<arguments>"

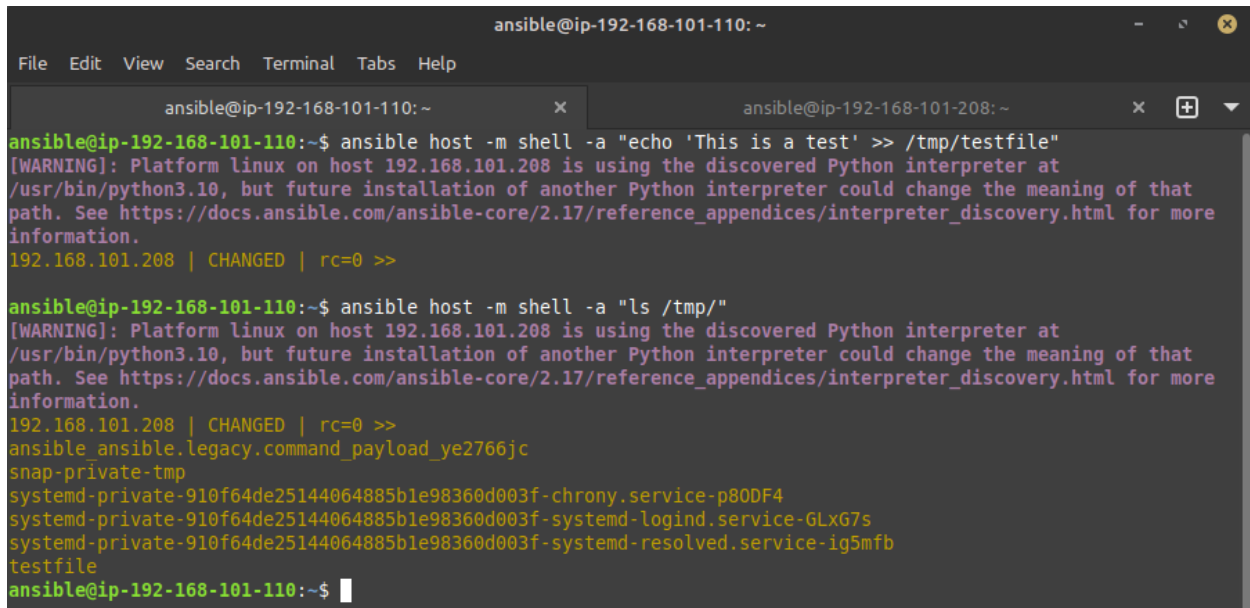
i. ansible all -m copy -a "src=/home/ansible/test.txt dest=/home/ansible/test.txt"



```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help

ansible@ip-192-168-101-110: ~
ansible@ip-192-168-101-110:~$ touch test.txt
ansible@ip-192-168-101-110:~$ ls
test.txt
ansible@ip-192-168-101-110:~$ ansible all -m copy -a "src=/home/ansible/test.txt dest=/home/ansible/test.txt"[W
ARNING]: Platform linux on host 192.168.101.208 is using the discovered Python interpreter at
/usr/bin/python3.10, but future installation of another Python interpreter could change the meaning of that
path. See https://docs.ansible.com/ansible-core/2.17/reference_appendices/interpreter_discovery.html for
more information.
192.168.101.208 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3.10"
  },
  "changed": true,
  "checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
  "dest": "/home/ansible/test.txt",
  "gid": 1001,
  "group": "ansible",
  "md5sum": "d41d8cd98f00b204e9800998ecf8427e",
  "mode": "0664",
  "owner": "ansible",
  "size": 0,
  "src": "/home/ansible/.ansible/tmp/ansible-tmp-1732555459.7214074-13178-36294721769125/.source.txt",
  "state": "file",
  "uid": 1001
}
```

ii. `ansible host -m shell -a "echo 'This is a test' >> /tmp/testfile"`



```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help

ansible@ip-192-168-101-110: ~ x  ansible@ip-192-168-101-208: ~ x + -
ansible@ip-192-168-101-110:~$ ansible host -m shell -a "echo 'This is a test' >> /tmp/testfile"
[WARNING]: Platform linux on host 192.168.101.208 is using the discovered Python interpreter at
/usr/bin/python3.10, but future installation of another Python interpreter could change the meaning of that
path. See https://docs.ansible.com/ansible-core/2.17/reference_appendices/interpreter_discovery.html for more
information.
192.168.101.208 | CHANGED | rc=0 >>

ansible@ip-192-168-101-110:~$ ansible host -m shell -a "ls /tmp/"
[WARNING]: Platform linux on host 192.168.101.208 is using the discovered Python interpreter at
/usr/bin/python3.10, but future installation of another Python interpreter could change the meaning of that
path. See https://docs.ansible.com/ansible-core/2.17/reference_appendices/interpreter_discovery.html for more
information.
192.168.101.208 | CHANGED | rc=0 >>
ansible_ ansible.legacy.command_payload_ye2766jc
snap-private-tmp
systemd-private-910f64de25144064885b1e98360d003f-chrorny.service-p80DF4
systemd-private-910f64de25144064885b1e98360d003f-systemd-logind.service-GLxG7s
systemd-private-910f64de25144064885b1e98360d003f-systemd-resolved.service-ig5mfb
testfile
ansible@ip-192-168-101-110:~$
```

# Ansible Playbook

An Ansible playbook is a YAML file that defines a series of tasks to be executed on host systems. It allows execution of series of task to multiple host in one go.

## Key Elements of a Playbook:

- **Target:** Part of playbook that defines host & basic configurations.
- **Tasks:** Each task defines a specific action to be performed. Tasks use Ansible modules to perform actions.
- **Variables:** You can define variables to customize the behavior of your playbook.
- **Handlers:** These are special tasks that only run when notified by another task. They are typically used for tasks like restarting services or reloading configurations after changes.

**Following is example of YAML ansible playbook**

```
---
- host: webservers
  become: yes # Run tasks with sudo privileges
  tasks:
    - name: Install nginx package
      apt:
        name: nginx
        state: present
    - name: Start nginx service
      service:
        name: nginx
        state: started
```

Playbook can be executed with following command:

```
>> ansible-playbook <playbook.yml>
```

## Ansible Playbook Examples:

i. Creating playbook01.yml which shows localhost's all the facts and variables:

```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help

ansible@ip-192-168-101-110: ~ x ansible@ip-192-168-101-208: ~ x +
GNU nano 6.2 playbook01.yml *
---
- hosts: localhost
  tasks:
    - name: Get local Information
      debug:
        var: hostvars[inventory_hostname]
```

## Running playbook:

>> ansible-playbook playbook01.yml

```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help

ansible@ip-192-168-101-110: ~ x ansible@ip-192-168-101-208: ~ x +
ansible@ip-192-168-101-110:~$ ansible-playbook playbook01.yml

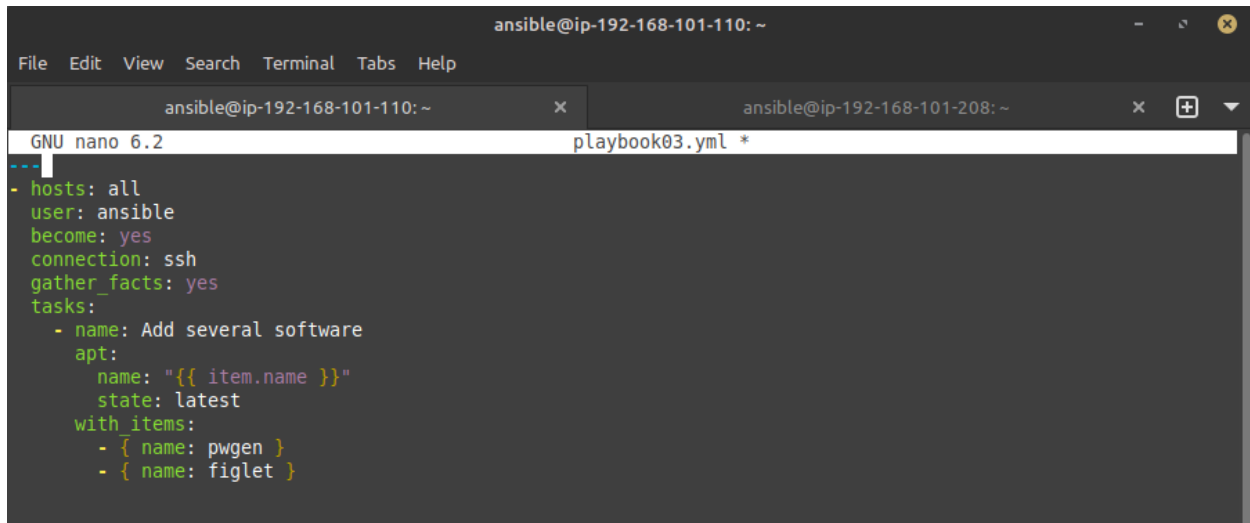
PLAY [localhost] *****
TASK [Gathering Facts] *****
ok: [localhost]

TASK [Get local Information] *****
ok: [localhost] => {
  "hostvars[inventory_hostname]": {
    "ansible_all_ipv4_addresses": [
      "192.168.101.110"
    ],
    "ansible_all_ipv6_addresses": [
      "fe80::88b:b1ff:fec8:ea83"
    ],
    "ansible_apparmor": {
      "status": "enabled"
    },
    "ansible_architecture": "x86_64",
    "ansible_bios_date": "10/16/2017",
    "ansible_bios_vendor": "Amazon EC2",
    "ansible_bios_version": "1.0",
    "ansible_board_asset_tag": "i-05dc28eda85153440",
    "ansible_board_name": "NA",
    "ansible_board_serial": "NA",
    "ansible_board_vendor": "Amazon EC2",
    "ansible_board_version": "NA",
    "ansible_chassis_asset_tag": "Amazon EC2",
    "ansible_chassis_serial": "NA",
    "ansible_chassis_vendor": "Amazon EC2",
    "ansible_chassis_version": "NA",
    "ansible_check_mode": false,
    "ansible_cmdline": {
      "BOOT_IMAGE": "/boot/vmlinuz-6.8.0-1015-aws",
      "console": "ttyS0",
      "nvme_core.io_timeout": "4294967295",
      "panic": "-1",
      "ro": true,
      "root": "PARTUUID=a645aa66-1c6e-48d7-ae9e-f8233e05b36c"
    },
    "ansible_config_file": "/etc/ansible/ansible.cfg",
    "ansible_connection": "local",
    "ansible_date_time": {
      "date": "2024-11-25",
      "day": "25",
      "epoch": "1732557592",
      "epoch_int": "1732557592",

```



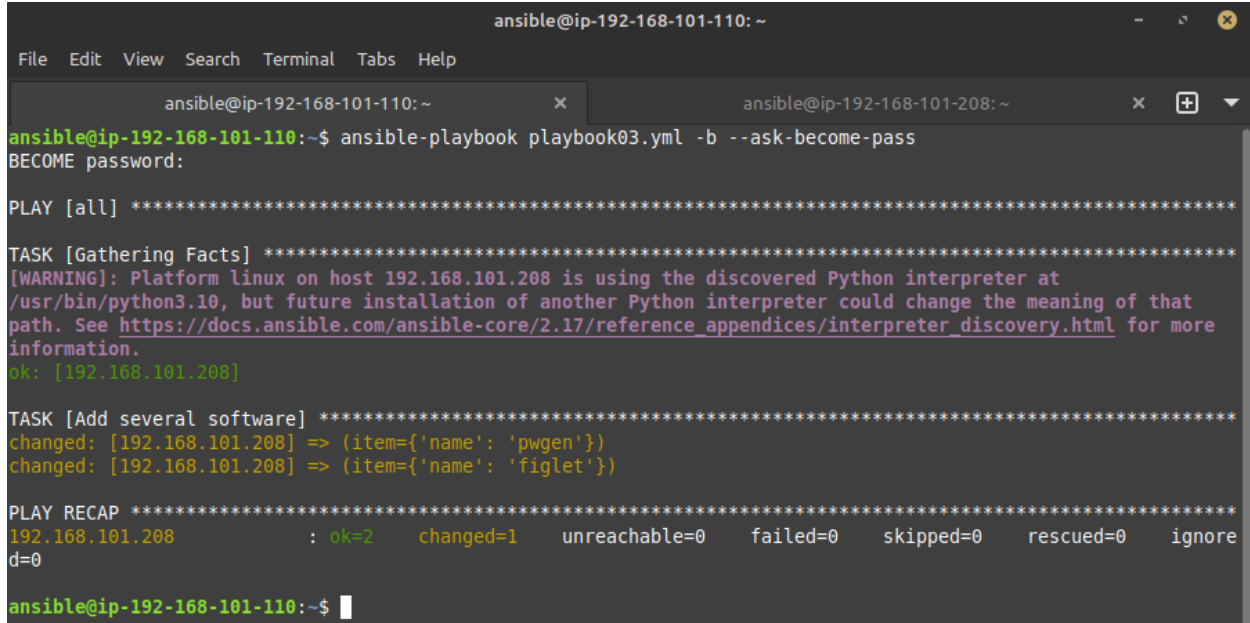
ii. Creating playbook03.yml which install/verify multiple apps on host system:



```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help
ansible@ip-192-168-101-110: ~ x ansible@ip-192-168-101-208: ~ x
GNU nano 6.2 playbook03.yml *
---
- hosts: all
  user: ansible
  become: yes
  connection: ssh
  gather_facts: yes
  tasks:
    - name: Add several software
      apt:
        name: "{{ item.name }}"
        state: latest
      with_items:
        - { name: pwgen }
        - { name: figlet }
```

Running playbook:

>> ansible-playbook playbook03.yml -b --ask-become-pass



```
ansible@ip-192-168-101-110: ~
File Edit View Search Terminal Tabs Help
ansible@ip-192-168-101-110: ~ x ansible@ip-192-168-101-208: ~ x
ansible@ip-192-168-101-110:~$ ansible-playbook playbook03.yml -b --ask-become-pass
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 192.168.101.208 is using the discovered Python interpreter at
/usr/bin/python3.10, but future installation of another Python interpreter could change the meaning of that
path. See https://docs.ansible.com/ansible-core/2.17/reference_appendices/interpreter_discovery.html for more
information.
ok: [192.168.101.208]

TASK [Add several software] *****
changed: [192.168.101.208] => (item={'name': 'pwgen'})
changed: [192.168.101.208] => (item={'name': 'figlet'})

PLAY RECAP *****
192.168.101.208      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignore
d=0

ansible@ip-192-168-101-110:~$
```

## Extra features of a Playbook:

### 1. Condition (when):

The when statement in playbook is used to execute a task only when a certain condition is met. The condition is typically a logical expression or variable value.

#### Example :

```
- name: Install nginx if the condition is true

ansible.builtin.yum:
  name: nginx
  state: present
  when: ansible_facts['os_family'] == 'RedHat'
```

### 2. Loops:

You can use the loop directive to iterate over a list of items and apply the same task for each item.

#### Example :

```
- name: Install multiple packages

ansible.builtin.apt:
  name: "{{ item }}"
  state: present
  loop:
    - nginx
    - git
    - vim
```

### 3. Tags:

Tags allow you to run specific parts of a playbook by assigning tags to tasks or plays. This helps you execute only certain parts of the playbook rather than running everything.

#### **Example :**

tasks:

- name: Install nginx

    ansible.builtin.yum:

        name: nginx

        state: present

tags:

- install

## Ansible Vault:

Ansible Vault is a feature that allows you to securely store and manage sensitive data, such as passwords, API keys, and private keys, within Ansible playbooks, inventory files, or variables. Vault ensures that this sensitive information is encrypted and can be easily decrypted when needed during playbook execution.

### How to Use Ansible Vault:

#### 1. Creating an Encrypted File

You can create an encrypted file using the `ansible-vault create` command. For example, to create an encrypted file called `secrets.yml`:

```
>> ansible-vault create secrets.yml
```

This command will prompt you for a password, which you will need to decrypt the file later.

#### 2. Viewing an Encrypted File

To view the contents of an encrypted file, use the `ansible-vault view` command:

```
>> ansible-vault view secrets.yml
```

#### 3. Editing an Encrypted File

If you need to edit an encrypted file, you can use the `ansible-vault edit` command:

```
>> ansible-vault edit secrets.yml
```

#### 4. Executing Encrypted File

To run the playbook that includes a vault-encrypted file, you would provide the vault password using the `--ask-vault-pass` option.

```
>> ansible-playbook --ask-vault-pass secrets.yml
```