

DATE : 08.04.2025

DT/NT : DT

LESSON : ANSIBLE

SUBJECT: PLAYBOOKS

BATCH : B 303

AWS-DEVOPS



TECHPRO
EDUCATION



techproeducation.com



+1 (585) 304 29 59





Table of Contents

- ▶ Intro to Playbooks
- ▶ Hosts and Users
- ▶ Tasks
- ▶ Modules
- ▶ Handlers
- ▶ Variables
- ▶ Conditionals and loops



Intro to Playbooks

- Playbooks are the basis for a really simple configuration management and multi-machine deployment system, unlike any that already exist, and one that is very well suited to deploying complex applications.
- Playbooks can declare configurations, but they can also orchestrate steps of any manual ordered process, even as different steps must bounce back and forth between sets of machines in particular orders.

```
name: Install and Configure MySQL
hosts: db-server
tasks:
  - name: Install Pre-Requisites
    yum: name=pre-req-packages state=present

  - name: Install MySQL Packages
    yum: name=mysql state=present

  - name: Start MySQL Service
    service: name=mysql state=started

  - name: Configure Database
    mysql_db: name=db1 state=present
```



Intro to Playbooks

- `ANSIBLE_CONFIG` (environment variable if set)
- `ansible.cfg` (in the current directory)
- `~/.ansible.cfg` (in the home directory)
- `/etc/ansible/ansible.cfg`

!!! `ansible.cfg` ✕

ansible4 > !!! `ansible.cfg`

```
1 [defaults]
2 host_key_checking = False
3 inventory = inventory.txt
4 deprecation_warnings=False
5 interpreter_python=auto_silent
6
```

* `vim hosts` `inventory`

```
[webservers]
node1 ansible_host=cnode1_ip> ansible_user=ec2-user
node2 ansible_host=cnode2_ip> ansible_user=ec2-user

[all:vars]
ansible_ssh_private_key_file=/home/ec2-user/<pem file>

[devservers]
node1

[testgroup:children]
webservers
devservers

[dev_sub_gr]
#node[1:20]
node[1:2]
```

Y `playbook.yml` ✕

ansible4 > Y `playbook.yml` > {} 0> {} tasks > {} 0> {} yum > E

```
1 ---
2 - name: db configuration
3   hosts: db_server
4   tasks:
5     - name: install mariadb and PyMySQL
6       become: yes
7       yum:
8         name:
9           - mariadb-server
10           - python3-PyMySQL
11         state: latest
12
```

control node



node1

node2

node3

node4

node5



Intro to Playbooks



Control
Node



Inventory



Playbook



Ansible



Hillroad Road, 43



Oxford Square, 15



Faringdon Road, 4



Roman Walk, 9





Intro to Playbooks

The goal of a play is to map a group of hosts to some well defined roles, represented by things ansible calls tasks. At a basic level, a task is nothing more than a call to an ansible module.



Intro to Playbooks

inventory.ini

```
[webservers]
node1 ansible_host=54.174.120.241
node2 ansible_host=3.84.254.65

[databases]
node3 ansible_host=54.174.102.205
```

play-1

play-2

playbook.yml

```
---
- name: update web servers
  hosts: webservers
  remote_user: root

  tasks:
  - name: ensure apache is at the latest version
    yum:
      name: httpd
      state: latest
  - name: write the apache config file
    template:
      src: /srv/httpd.j2
      dest: /etc/httpd.conf

- name: update db servers
  hosts: databases
  remote_user: root

  tasks:
  - name: ensure postgresql is at the latest version
    yum:
      name: postgresql
      state: latest
  - name: ensure that postgresql is started
    service:
      name: postgresql
```


Intro to Playbooks

inventory.ini

```
[webservers]
node1 ansible_host=54.174.120.241
node2 ansible_host=3.84.254.65

[databases]
node3 ansible_host=54.174.102.205
```

How to Run

```
$ ansible-playbook playbook.yml
```

modules

- Playbooks contain plays
- Plays contain tasks
- Tasks call modules
- Tasks run sequentially

playbook.yml

```
- name: update web servers
  hosts: webservers
  remote_user: root
```

tasks:

```
- name: ensure apache is at the latest version
```

```
  yum:
```

```
    name: httpd
    state: latest
```

} arguments

task-1

```
- name: write the apache config file
```

```
  template:
```

```
    src: /srv/httpd.j2
    dest: /etc/httpd.conf
```

} arguments

task-2

```
- name: update db servers
  hosts: databases
  remote_user: root
```

tasks:

```
- name: ensure postgresql is at the latest version
```

```
  yum:
```

```
    name: postgresql
    state: latest
```

} arguments

task-1

```
- name: ensure that postgresql is started
```

```
  service:
```

```
    name: postgresql
```

} arguments

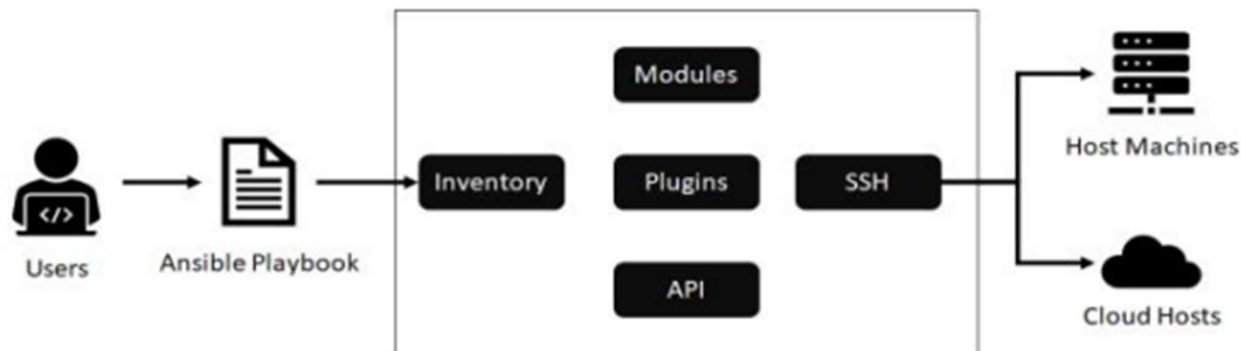
task-2

Ansible

A

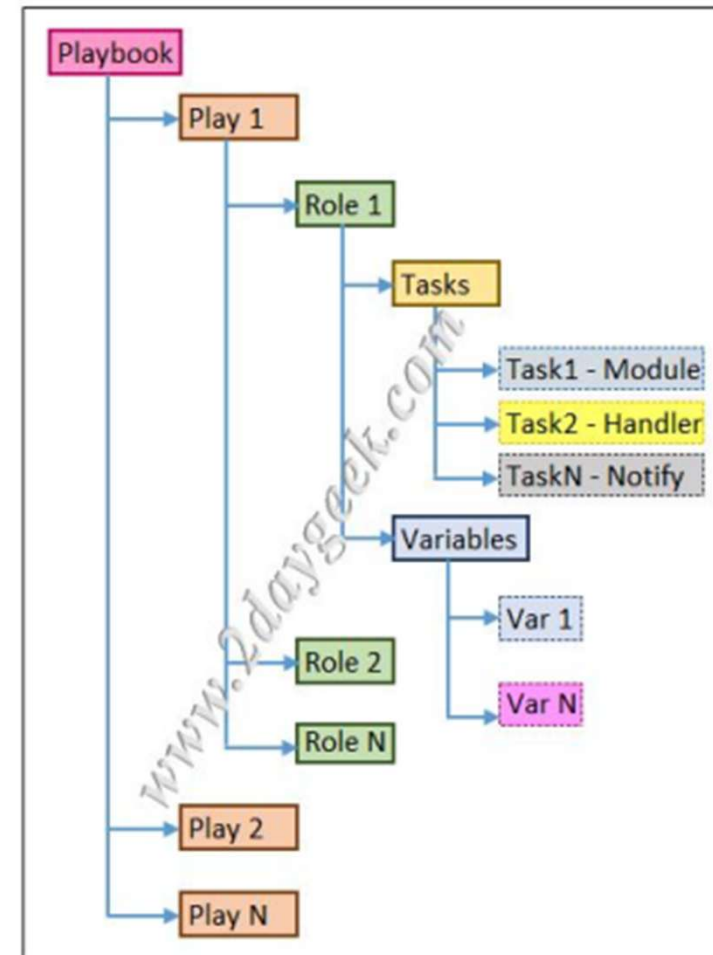
Hosts and Users

- For each play in a playbook, you get to choose which machines in your infrastructure to target and what remote user to complete the steps (called tasks) as.
- The host defined in the inventory file must match the host used in the playbook and all connection information for the host is retrieved from the inventory file.

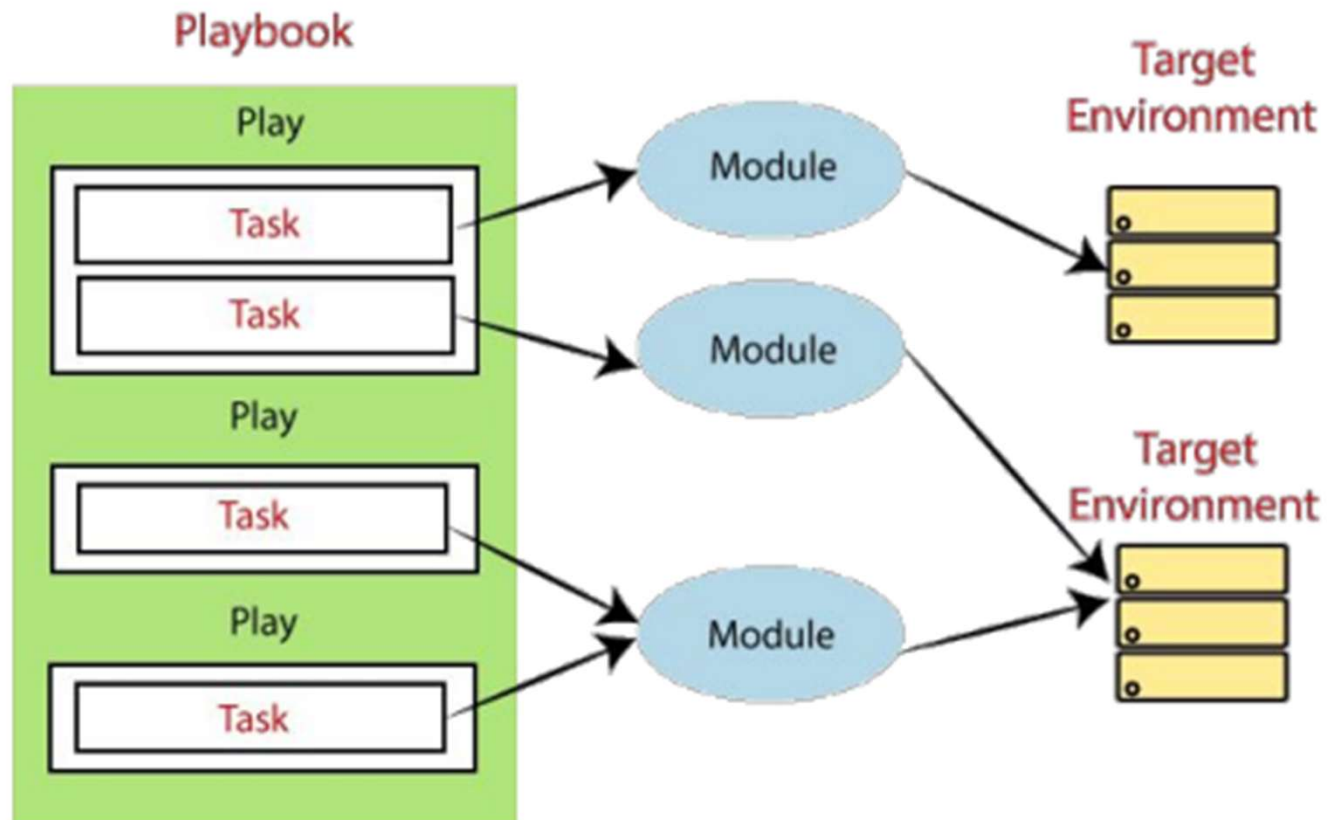


Tasks

- Each play contains a list of tasks. Tasks are executed in order, one at a time, against all machines matched by the host pattern, before moving on to the next task.
- The goal of each task is to execute a module, with very specific arguments. Variables can be used in arguments to modules.



Modules



Modules

- Modules (also referred to as “task plugins” or “library plugins”) are discrete units of code that can be used from the command line or in a playbook task.
- Ansible executes each module, usually on the remote target node, and collects return values.
- Modules should be **idempotent**, and should avoid making any changes if they detect that the current state matches the desired final state.

```
playbook.yml
-
  name: Play 1
  hosts: localhost
  tasks:
    - name: Execute command 'date'
      command: date
    - name: Execute script on server
      script: test_script.sh
    - name: Install httpd service
      yum:
        name: httpd
        state: present
    - name: Start web server
      service:
        name: httpd
        state: started
```

Handlers

- Handlers are lists of tasks, not really any different from regular tasks, that are referenced by a globally unique name, and are notified by notifiers. If nothing notifies a handler, it will not run.

```
- hosts: webserver1
  user: root
  tasks:
    - name: test copy
      copy: src=/root/a.txt dest=/mnt
      notify: test handlers
  handlers:
    - name: test handlers
      shell: echo "abcd" >> /mnt/a.txt
```





Inventory File

- Ansible works against multiple managed nodes or “hosts” in your infrastructure at the same time, using a list or group of lists known as inventory.
- The default location for inventory is a file called `/etc/ansible/hosts`.
- You can specify a different inventory file at the command line using the `-i < path >` option.



Inventory Files

```
$ app.inv
[webservers]
www1.example.com
www2.example.com

[appservers]
app1.example.com
app2.example.com

[memcached]
memcached.example.com

[redis]
redis.example.com

[dbservers]
db0.example.com
```

Variables

- **Variables** are used to store values that varies with different items.

```
[webservers]
web1 ansible_host=3.85.110.235 ansible_user=ec2-user ansible_ssh_pass=P@abcd
web2 ansible_host=3.88.62.253 ansible_user=ec2-user ansible_ssh_pass=P@1234

[dbservers]
db1 ansible_host=3.85.110.235 ansible_user=ec2-user ansible_ssh_pass=P@Defne
```

Playbook.yml

```
-
  name: Add DNS server to resolv.conf
  hosts: webservers
  vars:
    dns_server: 10.1.250.10
  tasks:
    - lineinfile:
      path: /etc/resolv.conf
      line: 'nameserver {{ dns_server }}'
```

OR

```
#Sample variable file - web.yml
dns_server: 10.1.250.10
```



Conditionals

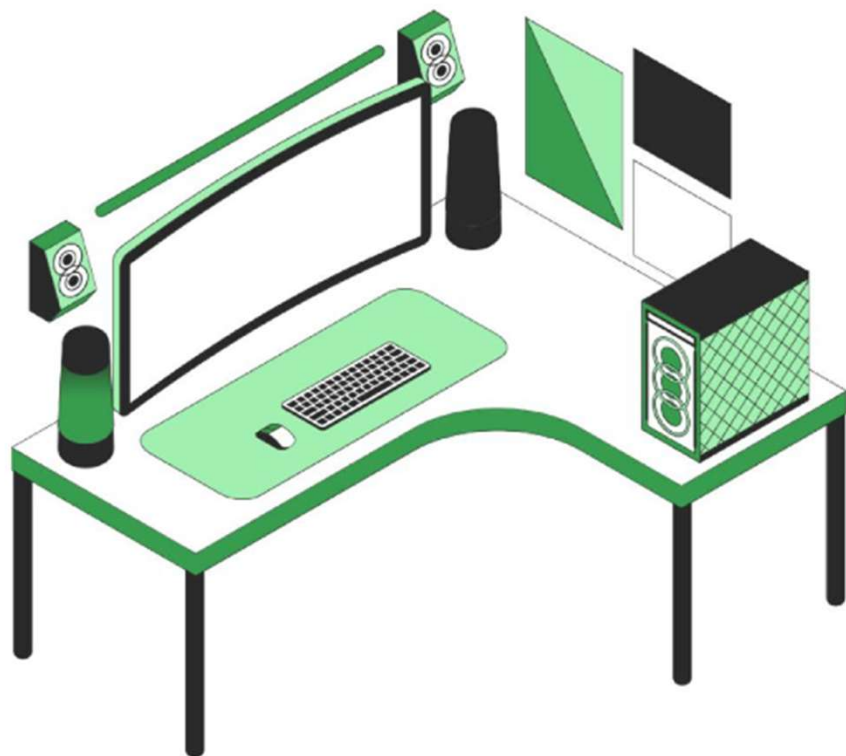
```
- name: Install NGINX
hosts: webserver
tasks:
- name: Install NGINX on Redhat
  yum:
    name: nginx
    state: present
    when: ansible_os_family == "RedHat"

- name: Install NGINIX on Debian
  apt:
    name: nginx
    state: present
    when: ansible_os_family == "Debian" and ansible_distribution_version == "16.04"
```

Loops

```
name: 'Install required packages'
hosts: webserver
tasks:
  -
    yum:
      name: '{{ item }}'
      state: present
      loop:
        - httpd
        - binutils
        - glibc
        - sysstat
        - unixODBC
        - mongodb
        - nodejs
        - grunt
```





Do you have any questions?

Send it to us! We hope you learned something new.