Ansible Playbooks

CLARUSWAY©
WAY TO REINVENT YOURSELF

Table of Contents

CLARUSWAY©
WAY TO REINVENT YOURSELF

1

Playbooks

Playbooks

Playbooks

- Plain-text **YAML** files that defines a set of activities (tasks) to be run on hosts.
- Human and machine readable.
- Can be used to build and configure entire application environments.

```
---
- 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
      state: started
```

Playbooks

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
      state: started
```


Playbooks

playbook.yml

play-1

```
---
- 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
```

play-2

```
- 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
      state: started
```

Playbooks

playbook.yml

play-1

```
---
- 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
```

play-2

```
- 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
      state: started
```


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
```

playbook.yml

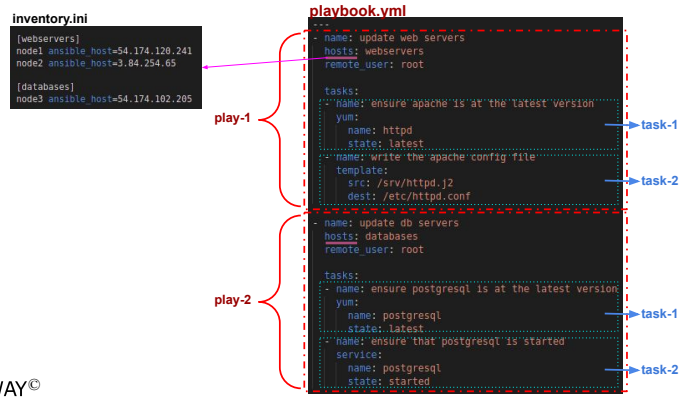
play-1

```
---
- 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
```

play-2

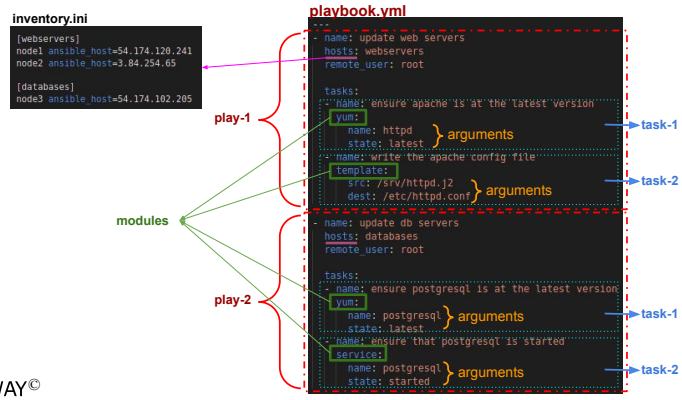
```
- name: update db servers
  hosts: databases
  remote_user: root

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

Playbooks

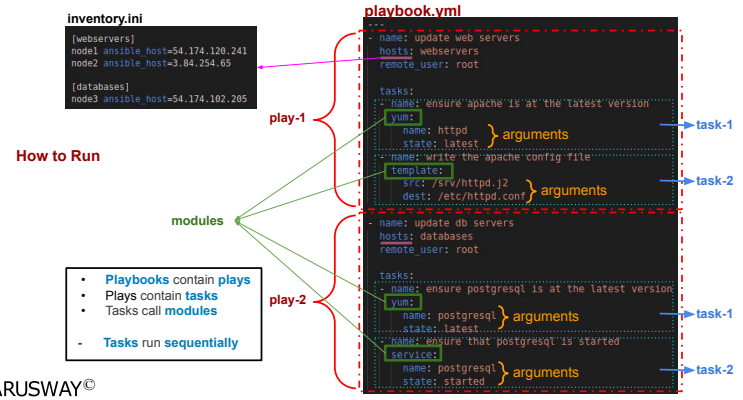


Playbooks

Playbooks



Playbooks

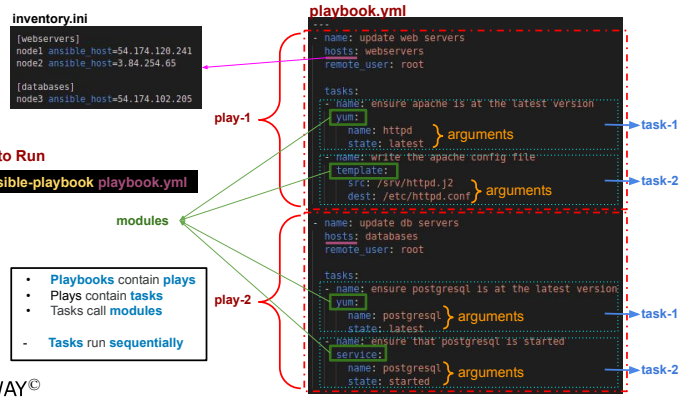


How to Run

- **Playbooks** contain **plays**
- Plays contain **tasks**
- Tasks call **modules**

- **Tasks** run **sequentially**

Playbooks



CLARUSWAY©
WAY TO REINVENT YOURSELF

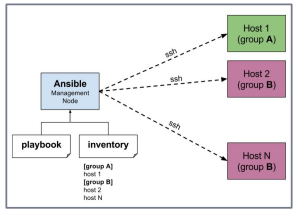
2

Hosts and Users

CLARUSWAY©
WAY TO REINVENT YOURSELF

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.




```yaml
---
- 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
```


3

Inventory File

Inventory File

- Ansible works against multiple managed nodes or “hosts” in your infrastructure at the same time, using a list or group of lists know 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.ini

```
54.174.120.241

mail.example.com

[webservers]
node1 ansible_host=54.174.120.241 ansible_user=root ansible_ssh_pass=PQabc
node2 ansible_host=3.84.254.65    ansible_user=ec2-user

[databases]
node3 ansible_host=54.174.102.205 ansible_user=root ansible_ssh_pass=PQabc

[dev]
node1
node3

[newyork]
node2
node3
```


4

Tasks

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.

```
# Simple Ansible Playbook1.yml
-
 name: Play 1
 hosts: localhost
 tasks:
   - name: Execute comand "date"
     command: date

   - name: Execute script on server
     script: test.sh

   - name: Install httpd package
     yum:
         name: httpd
         state: present

   - name: Start web server
     service:
         name: httpd
         state: started
```


5 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
```


6 Handlers

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: webservers1
  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
~
```


7 Variables

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_shh_pass=P@abcd
web2 ansible_host=3.88.62.253 ansible_user=ec2-user ansible_shh_pass=P@1234

[dbservers]
db1 ansible_host=3.85.110.235 ansible_user=ec2-user ansible_shh_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 - webservers.yml

dns_server: 10.1.250.10
```


8 Conditionals

Conditionals

```yaml
- name: Install NGNIX
  hosts: webservers
  tasks:
  - name: Install NGNIX on Redhat
    yum:
      name: ngnix
      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"
```


9 Loops

Loops

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


THANKS!

Any questions?