Ansible - Configuration management tool

By, Ganesh Palnitkar

Agenda

- What is Ansible?
- Installing Ansible
- Ansible Playbooks
- Using ansible for Configuration management tasks

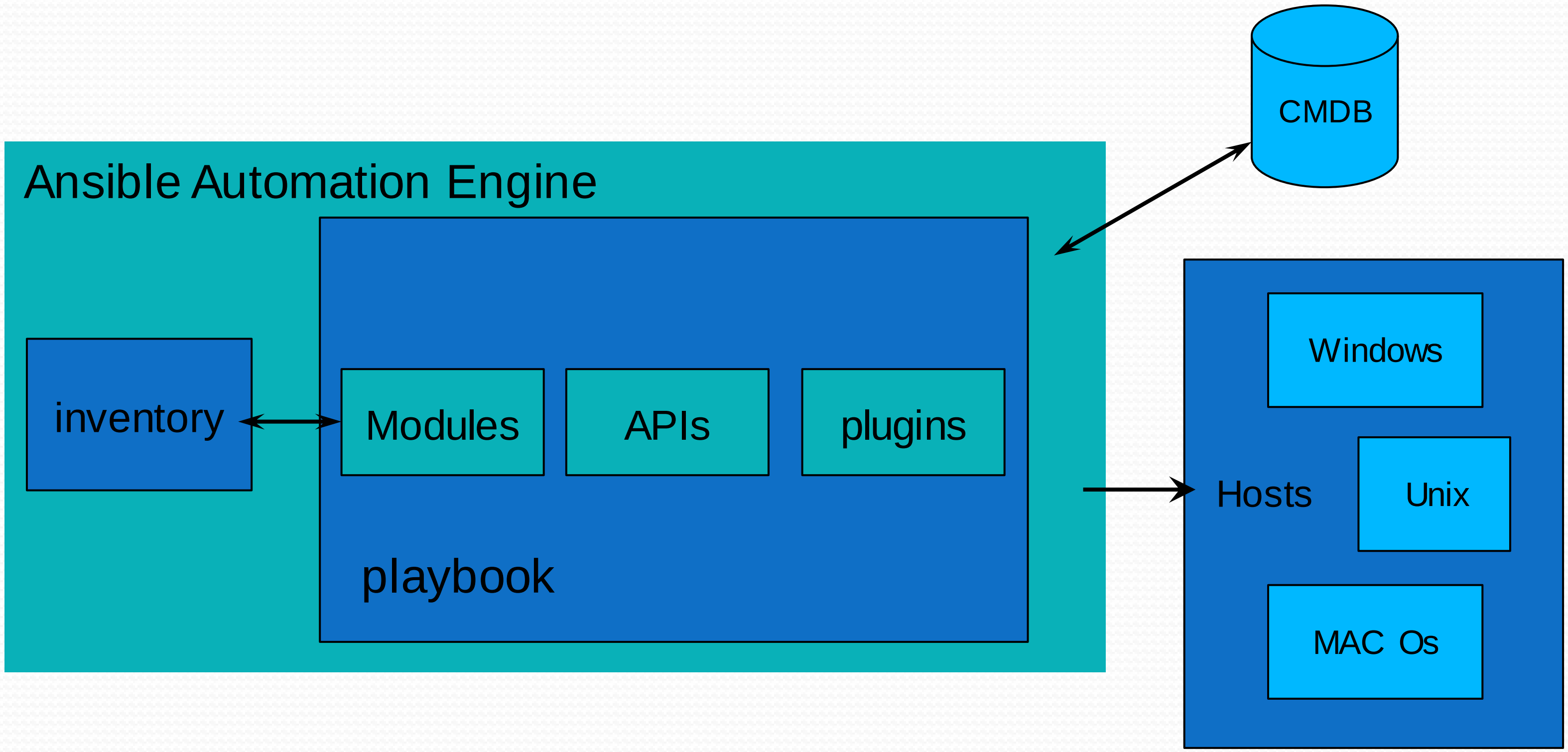
What is Ansible

- Ansible is an **automation engine** that is used as configuration management tool, deployment tool, provisioning tool and virtually automate many of infrastructure processes.
- **Agentless operation!**
- Ansible uses **Python** scripting / programs for calling playbook and other code inside Ansible. This means that you need to have Python version 2.5+ installed on the control server and nodes that you want to control.
- Ansible **Inventory file:** (/etc/ansible/hosts)

```
[webservers]
webserver1.example.com
webserver2.example.com
[dbservers]
dbserver1.example.com
dbserver2.example.com
```

- An **inventory** file can be created **dynamically** using python script like ec2.py and similar ones from google cloud etc.

Ansible Architecture



About Ansible

- Ansible use **SSH** for communicating with nodes, thus ssh protocol should be enabled on the nodes and server. The nodes must have **python** installed on them in order to run the playbook.
- The Playbooks are expressed in YAML language (*.yaml extension file)
- Some **ad-hoc commands** that can be fired from the Ansible server.
  - $ ansible all –m ping
  - $ ansible webservers –m yum –a “name=httpd state=installed”
  - $ ansible webservers -a “usr/sbin/reboot”
- Ansible has **built-in modules** that can be used for various operation like package installation, deployment, installing a particular app.
- In order to have more structured way to manage your infrastructure we can create and use Ansible Playbooks. Below is a sample playbook that can be used for infra config management.
- http://docs.ansible.com/ansible/modules_by_category.html ---- list of module available on Ansible archive.

Installing Ansible

Installing Ansible on Linux variants:

- Debian/Ubuntu: The easiest way to install Ansible on a Debian or Ubuntu system is to use the official apt package.

```
$ sudo apt-add-repository -y ppa:ansible/ansible
$ sudo apt-get update
$ sudo apt-get install -y ansible
```

- Incase there's an error about Python, run below command.

```
$ sudo apt-get install python-software-properties
```

- Once installed check version by running below command,

```
$ ansible –version
```

- Create a file at /etc/ansible/hosts (the default location for Ansible's inventory file), and add one server to it:

```
$ sudo mkdir /etc/ansible
$ sudo touch /etc/ansible/hosts
```

- Edit this hosts file with nano, vim, or whatever editor you'd like, but note you'll need to edit it with sudo as root. Put the following into the file:

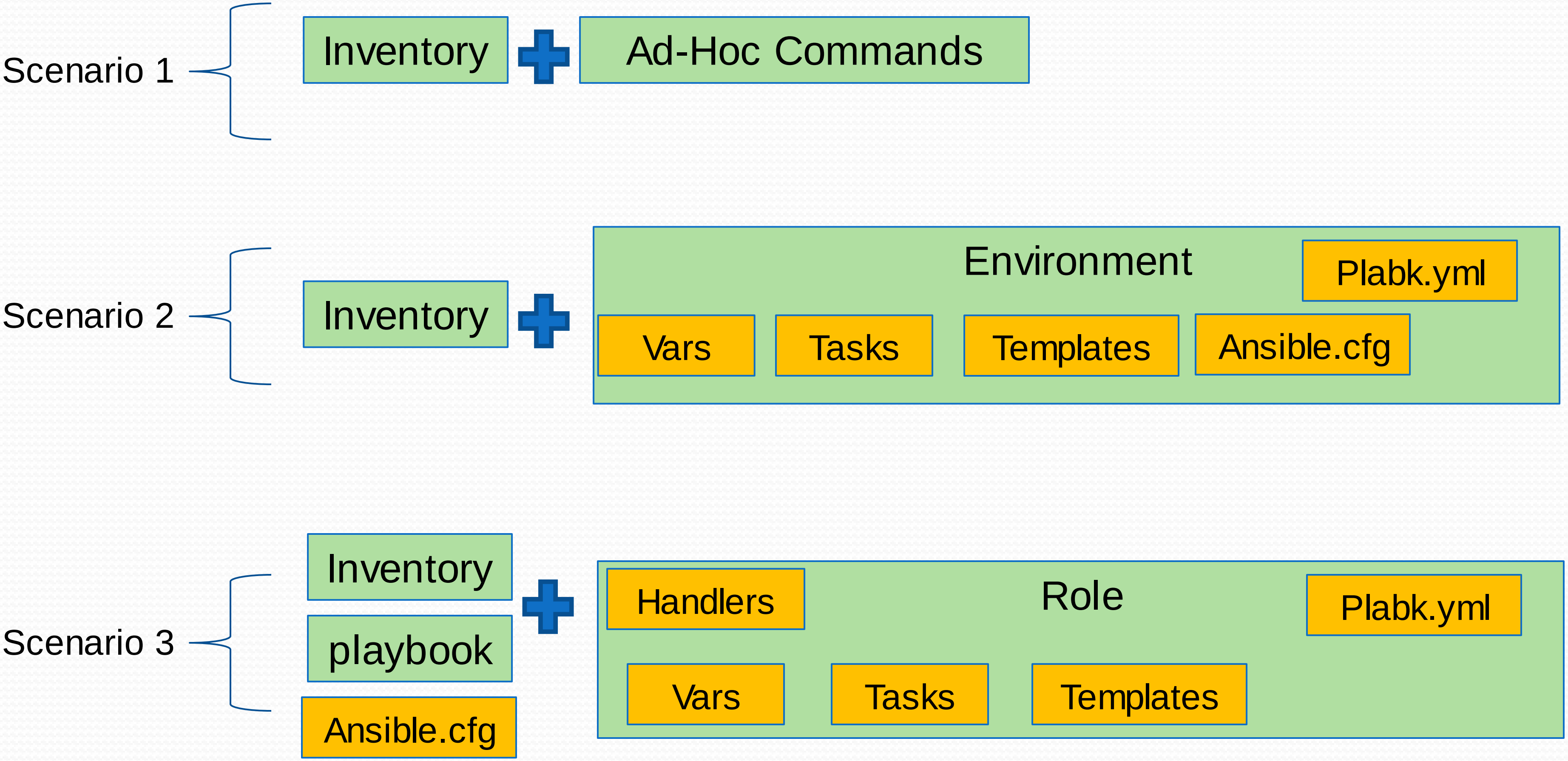
```
[webserver]
  webserver1
  ……
```

Preparing for Windows connection

The **Ansible** server should be installed with some of the required Python packages that are required for communicating with **Windows** machine.

- Install below python package one-by-one using the over command line,
    - $ pip install markupsafe
    - $ pip install xmltodict
    - $ pip install pywinrm

Ansible Lab Setup



Scenario 1

Inventory + Ad-Hoc Commands

Scenario 2

Inventory +

Environment

Plabk.yml

Vars

Tasks

Templates

Ansible.cfg

Scenario 3

Inventory

playbook

Ansible.cfg

+

Handlers

Role

Plabk.yml

Vars

Tasks

Templates

Ansible Inventory

Ansible inventory is a collection of server / nodes that we want to control using the Ansible CF tool.

There are two types in this to maintain the inventory,

- Static inventory
- Dynamic inventory

Static inventory is typically declared in /etc/ansible/hosts file. This location can be altered as per requirement and should be mentioned accordingly in 'ansible.cgf' file.

```
[dbserver]
mysql1.example.com
[webserver]
webserver.example.com
[appserver]
appserver [01:25].example.com
```

$ ansible –i /etc/ansible/hosts –m ping dbserver - - - - - inventory mentioned in 'hosts' under the group 'dbserver' are ping using the 'ping' module

Incase there are multiple servers inside a group one can only perform action on selected server inside a group by using the '- l' limit switch.

Dynamic inventory is generated / updated using script written in python or shell by querying the AWS remote server environment. AWS provides a python script (ec2.py). Ansible provides plugins (script) for most of the cloud environments, like Google, Azure, AWS, etc.

Ansible Module

Ansible module are available for almost all types of work that one would want to carry out.
On ansible server these can be listed using command,

```
$ ansible-doc –l
```

Also more information can retrieved using command,

```
$ ansible-doc <module_name>
```

Module Category

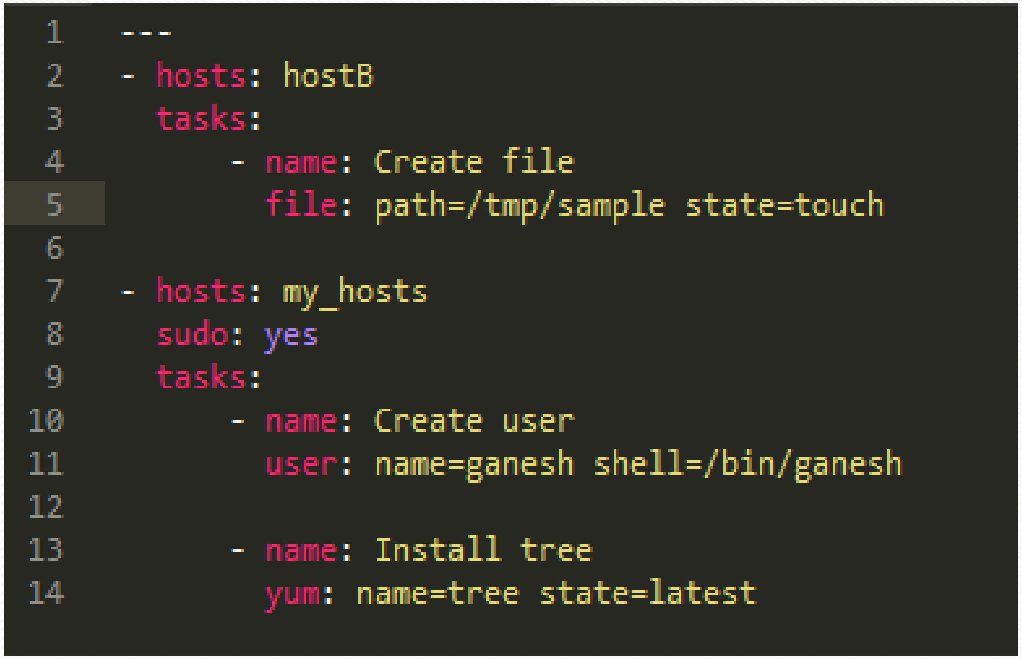
1) Manage server
2) Maintain virtual servers
3) Configure network equipment.
4) Maintain database and tables
5) Deploy load balancer configuration.

Example the ‘setup’ module fetch the remote hosts profile information. Thus the setup module actually runs the ‘Ohai’ profiler.

Ansible playbook

- The **syntax** of writing the playbook is very specific. Specially about **indentation.**
- **Running** a ‘playbook’ from command line is done using below command,

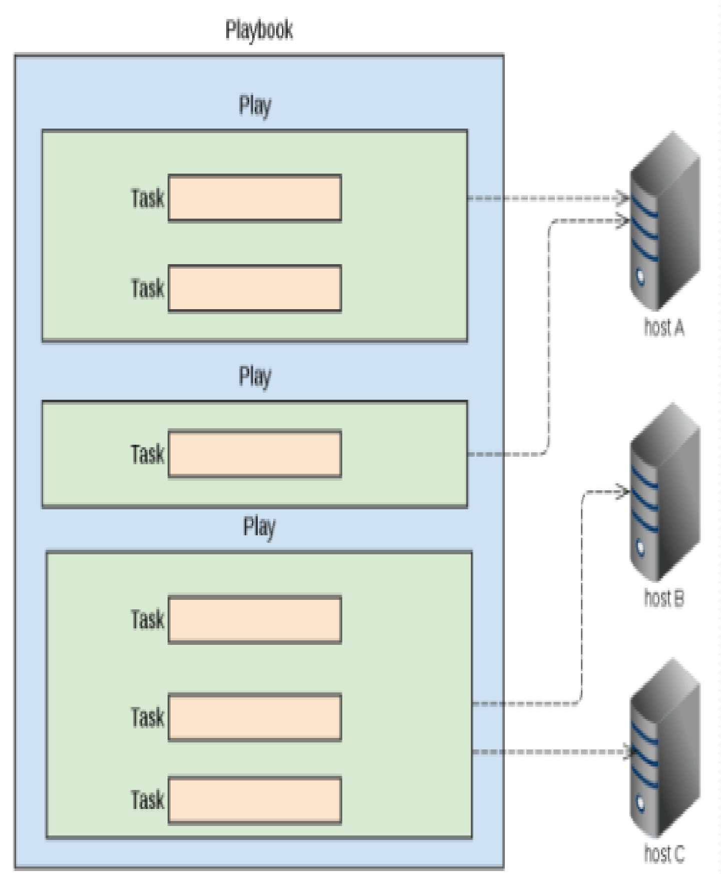
  $ ansible-playbook <xyz-playbook.yml.>



```yaml
---
- hosts: hostB
  tasks:
      - name: Create file
        file: path=/tmp/sample state=touch

- hosts: my_hosts
  sudo: yes
  tasks:
      - name: Create user
        user: name=ganesh shell=/bin/ganesh

      - name: Install tree
        yum: name=tree state=latest
```



Ansible Playbook

```
---
- hosts: webservers
  vars:
     http_port: 80
     max_clients: 200
  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
    notify:
      - restart apache
    - name: ensure apache is running (and enable
it at boot)
      service: name=httpd state=started
enabled=yes
  handlers:
    - name: restart apache
      service: name=httpd state=restarted
```

```
---
-  hosts: testgroup1
   remote_user: root

   tasks:
     - name: ensure apache is at the latest
version
       apt: name=apache2 state=latest
     - name: write the apache config file
       template: src=/srv/apache2.j2
dest=/etc/apache2.conf

-   hosts: testgroup2
    remote_user: root

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

- Tasks execution happens in order they are written.
- Ansible has built-in module for creating EC2 instance on Amazon cloud. Also available for other types of cloud environments.

Ansible Roles

Ansible roles help to ease the maintenance and troubleshooting of Ansible Playbooks.

Structure of Ansible roles:

A role would typical contain below objects,

- files – usually the files that are to be copied to the target servers.
- host_vars / group_vars - variable definitions.
- meta – dependencies in a role.
- templates – files that contain dynamic data.
- tasks – mainly actions statements in .yml file.
- handlers – to perform dependent action to be performed.

Ansible Tower

Web based access to configuration management.

Ansible tower dashboard provides information about,

1) Number of hosts
2) Inventories
3) Failed hosts
4) Projects, project synch failures
5) Graphical representation of Job status.

Ansible Tower is available as a evaluation version for a limited period of time. This can be downloaded from Ansible website from three resources, namely.

1) Download package as per your Linux / Unix variant.
2) Launch Ansible Tower on Vagrant
3) Launch Tower in Amazon EC2.

Thank You