Ansible Fundamentals

By: Shariq Akhtar

Whenever you need to run certain tasks like provisioning or maintaining a resource in your infrastructure or deploying something on web servers and similar kind of repetitive tasks, it becomes tedious and boring, plus we could make mistakes if we're doing it manually. You know we're humans! That's when Ansible comes to your rescue.

In this article we will take a look at Ansible fundamentals, like, what is Inventory, Task, Module and a Playbook. Let's start our simple and short journey with what Ansible is all about.

Ansible is open-source IT automation and orchestration tool backed by Red Hat. That let you run the tasks to manage your configuration for software provisioning on different resources. It is easy and simple to use. Other configuration management tools are Chef, Puppet, Salt and so on.

Ansible is popular due to the following reasons.

- Easy Setup.
- Easy Management.
- High Scalability.
- Inexpensive in comparison to others.
- Agentless.

Some of the positives of using Ansible are:

- It reduces human error.
- It is easily & quickly scalable.
- It automates application deployment.
- It helps in continuous delivery.

It is Agentless, hmmmm! What does it mean? It requires no additional installable dependencies to function by itself.

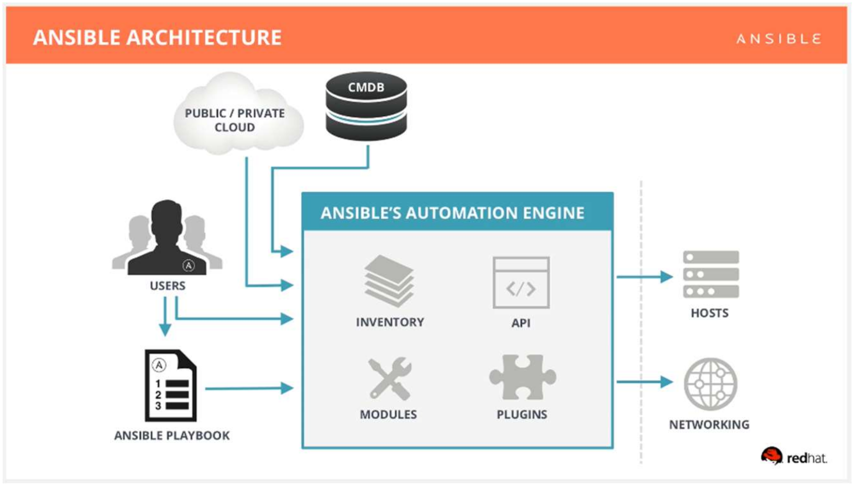
Ansible automation runs in YAML based which is easy and human readable. **Playbooks** are written in this language. Playbooks carry out the process for automation.

Inventory is a host or resources list that could have a categorized or grouped into list of hosts. It could be static list of servers or a dynamic list of servers from your favorite cloud provider like AWS, Azure, GCP or any other.

As mentioned on Ansible.com, Playbooks contain **Plays**. Plays contain **Tasks**. Tasks call **Modules.** Tasks run sequentially. **Handlers** are triggered by tasks and are run once, at the end of the plays.

Modules are programs that can run all the actions that you would want to in order to complete your tasks e.g. installing apache server, installing node, installing a library, creating a file and so on. Ansible has over 450 modules to choose from.

Below is the Ansible architecture diagram from Red hat. User runs the playbook on controller machine, playbook accesses the inventory of resources, run module and plugin and API is used, if there is any cloud API is required to access resources. Then the required tasks are performed for the targeted hosts.



Credits: Red Hat Ansible

You could make an inventory in different formats like .py, .yml, .ini and so on. Most popular is YAML or INI. Below is a sample Inventory list of hosts in a file.

```
[webservers]
web-server-1.com
web-server-2.com

[dbservers]
db-server-1.com
db-server-2.com
```

Let's get back to the playbook and examine our playbook file, which is in YAML format. The purpose of this playbook is to install and start our nginx on a host in or inventory. First we will write the file and then run it to complete our tasks.

File: playbook.yml

```
---
-  name: install nginx and start nginx service
   hosts: all
   become: true

   tasks:
   -  name: install latest nginx
      yum:
        name: nginx
        state: latest

   -  name: start nginx
      yum:
        name: nginx
        state: started
```

If we create our file in a good editor with YAML extensions our file is good, otherwise if we want to validate our file we can use YAML Lint or other online validators.

Now let us run our playbook. Once we are in our host machine and that has ansible installed on it, we are going to use the bellow command to run our playbook.

```
[user@some-anisble-controller]$ ansible-playbook -i hosts playbook.yml
```

In this case our hosts file has one single entry like so.

```
172.31.17.17
```

This will install nginx on 172.31.17.17 host and also start the nginx service. To verify that we can open up the browser and check the port 80 for nginx.

It should show us welcome screen by nginx. Another way is to use `ngnix -v` on command line.

So that's all guys! This is all you need to get started and off course we can do much more in Ansible as we get our hands dirt and dig deeper, we can truly leverage some of the key features of Ansible in our day-to-day repetitive tasks and make our CI/CD process automated.