Ansible 101

How to automate your system administration tasks with Ansible

Jeff Smith: crankyoldbugger@gmail.com

What is Ansible?

The Ansible website explains the project as "a radically simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration, and many other IT needs." Ansible can be used to perform the same tasks across a defined set of servers from a centralized location.

What Ansible Can Do?

- **Configuration Management** - The enterprise hardware and software information is recorded and updated in detail, thus maintaining the consistency of the product performance.
- **Application Deployment** - The applications can be managed in Ansible from Development to Production when you define and manage the applications using Ansible.
- **Orchestration** - To manage as a whole and how the configurations interact.
- **Security and Compliance** - Wide security policy can be deployed across the infrastructure when the policy is defined in Ansible
- **Provisioning** - Helps to automate and manage the process

How does Ansible work?

- Ansible translates Ansible playbooks into commands that are run over SSH, which has several benefits when it comes to managing Unix-like environments:

  - Most, if not all of the Unix-like machines you are administering will have SSH running by default.
  - Ansible uses SSH protocol to connect to servers and run tasks. By default, Ansible uses SSH keys with ssh-agent and connects to remote machines using your current user name. Root logins are not required. You can log in as any user, and then su or sudo to any user.
  - In most cases no additional software needs to be installed as Ansible requires Python 2.6 in order to operate. Most, if not all distributions of Linux have this version (or greater) installed by default.
  - Ansible does not require a master node. It can be run from any host that has the Ansible package installed and sufficient SSH access.
  - Although running Ansible in a cron job is possible, by default Ansible only runs when you tell it to.

Idempotent (/ˌɪd.əmˈpəʊ.tənt/)

- If you are familiar with Bash for-loops, you'll find that Ansible operates in a similar fashion.
- The difference, however, is that Ansible is idempotent. In layman's terms this means that generally Ansible only performs the requested action if a change will occur as a result.
- Most Ansible modules check whether the desired final state has already been achieved, and exit without performing any redundant actions, so that repeating the task does not change the final state. Modules that behave this way are often called ‘idempotent.’

Ansible Galaxy

- Galaxy is a hub for finding and sharing Ansible content.
- Galaxy provides pre-packaged units of work known to Ansible as roles and collections.
- Content from roles and collections can be referenced in Ansible PlayBooks and immediately put to work. You'll find content for provisioning infrastructure, deploying applications, and all of the tasks you do everyday.
- Use the Search page to find content for your project, then download them onto your Ansible host using ansible-galaxy, the command line tool that comes bundled with Ansible.

  Ansible Galaxy can be found at https://galaxy.ansible.com

Ansible Automation Platform (formerly Ansible Tower)

- The control plane for Ansible Automation Platform is the automation controller (replacing Ansible Tower). It includes a user interface (UI), role-based access control (RBAC), workflows, and continuous integration and continuous delivery (CI/CD) for helping your team scale with more efficiency and flexibility.
- Automation controller helps standardize how automation is deployed, initiated, delegated, and audited, allowing enterprises to automate with confidence while reducing sprawl and variance. Manage inventory, launch and schedule workflows, track changes, and integrate into reporting, all from a centralized user interface and RESTful API.

The Hosts File

- Location of your ansible hosts file can be set in /etc/ansible/ansible.cfg
- Can be in yaml format, or as “normal” text.

Yaml format

```
all:
  children:
    dev:
      hosts:
        devsrv01:
        devsrv02:
    prod:
      hosts:
        prodsrv01:
        prodsrv02:
        prodsrv03:
    webservers:
      hosts:
        websrv01:
        websrv02:
```

Normal format

```
[dev]
devsrv01
devsrv02

[prod]
prodsrv01
prodsrv02
prodsrv03

[webservers]
websrv01
websrv02
```

Playbooks and Roles

- Playbooks and roles are very similar, yet very different at the same time. A playbook is a standalone file that Ansible can run that contains all of the information required to set a machine’s state to what you expect. This means that a playbook can contain variables, tasks, handlers, roles, and even other playbooks, all in the same file. You don’t need any other files to accomplish your task.
- You can think of a role as a playbook that is split up into multiple different files. Instead of having a single file that contains everything that you need, there’s one file for variables, one for tasks, and another for handlers. You can’t run a role on its own, though; you need to include it inside a playbook along with the information about which hosts to run on.

Intro to Playbooks

- Ansible Playbooks offer a repeatable, re-usable, simple configuration management and multi-machine deployment system, one that is well suited to deploying complex applications. If you need to execute a task with Ansible more than once, write a playbook and put it under source control. Then you can use the playbook to push out new configuration or confirm the configuration of remote systems. The playbooks in the ansible-examples repository illustrate many useful techniques. You may want to look at these in another tab as you read the documentation.
- Playbooks can:
  - declare configurations
  - orchestrate steps of any manual ordered process, on multiple sets of machines, in a defined order
  - launch tasks synchronously or asynchronously

A Sample Playbook

```
---
- hosts: all
  gather_facts: True
  become: true
  strategy: free
  tasks:

  - name: Enable EPEL Repository on CentOS 8
    yum:
      name: epel-release
      state: latest
    when: ansible_facts['os_family'] == 'RedHat' and ansible_facts ['distribution_major_version'] == '8'
```

Links

- links: all

Beginner Tutorial: https://linuxconfig.org/ansible-tutorial-for-beginners-on-linux

Playbook Keywords: https://docs.ansible.com/ansible/latest/reference_appendices/playbooks_keywords.html

Jeff Geerling Videos: https://www.youtube.com/watch?v=goclfp6a2IQ

Free Training from Red Hat: https://www.redhat.com/en/services/training/do007-ansible-essentials-simplicity-automation-technical-overview?utm_medium=Email&utm_campaign=weekly&sc_cid=7013a000002pnItAAI

An Ansible Reference Guide: https://www.digitalocean.com/community/cheatsheets/how-to-use-ansible-cheat-sheet-guide

Automate VM Deployment: https://www.redhat.com/sysadmin/deployment-ansible-design

Using Ansible to automate LVM: https://www.redhat.com/sysadmin/automating-logical-volume-manager