**INTRODUCTION OF ANSIBLE:**

**Ansible** is an open-source automation engine that automates software provisioning, configuration management, and application deployment. Ansible lets you control and configure nodes from a single machine. What makes it different from other management software is that Ansible uses SSH infrastructure. Ansible uses push mode, where the configuration is pushed from a master machine to nodes.

* It supports configuration management with examples as below.
* Configuration of servers
* Application deployment
* Continuous testing of already install application
* Provisioning
* Orchestration
* Automation of tasks

**WHY ANSIBLE?**

Working in IT, you're likely doing the same tasks over and over. What if you could solve problems once and then automate your solutions going forward? Ansible is here to help.

It is included as part of the [Fedora](https://en.wikipedia.org/wiki/Fedora_(operating_system)) distribution of Linux, owned by [Red Hat Inc.](https://en.wikipedia.org/wiki/Red_Hat), and is also available for [Red Hat Enterprise Linux](https://en.wikipedia.org/wiki/Red_Hat_Enterprise_Linux), [CentOS](https://en.wikipedia.org/wiki/CentOS), and [Oracle Linux](https://en.wikipedia.org/wiki/Oracle_Linux) via Extra Packages for Enterprise Linux (EPEL) as well as for other operating systems.

Ansible, Inc. (originally Ansible Works, Inc.) was the company set up to commercially support and sponsor Ansible [Red Hat](https://en.wikipedia.org/wiki/Red_Hat) acquired Ansible in October 2015.

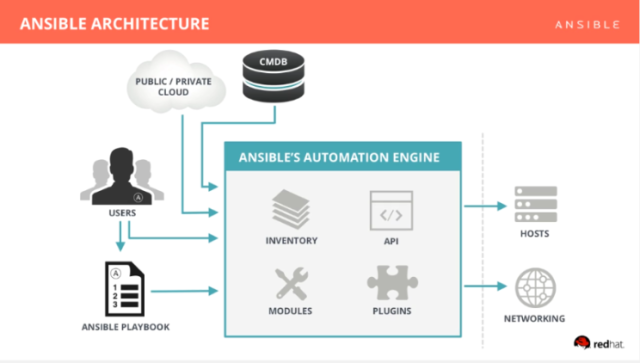
**WHEN YOU AUTOMATE, YOU ACCELERATE**

* Team impact
  + Save time and be more productive
  + Eliminate repetitive tasks
  + Fewer mistakes and errors
  + Improve collaboration job satisfaction
* Enterprise impact
  + Overcome complexity
  + More resources for innovation
  + Increase accountability and compliance

**ANSIBLE IS**

* Simple
  + Human readable automation
  + No special coding skills needed
  + Tasks executed in order
  + **Get productive quickly**
* Powerful
  + App deployment
  + Configuration management
  + Workflow orchestration
  + **Orchestrate the app lifecycle**
* Agentless
  + Agentless architecture
  + Uses OpenSSH and WinRM
  + No agents to exploit or update
  + **Predictable, reliable and secure**

**ARCHITECTURE OF ANSIBLE:**



Ansible has two types of servers: controlling machines and nodes. First, there is a single controlling machine which is responsible for managing the nodes over SSH. The controlling machine describes the location of nodes through its inventory files.

Ansible uses an agentless architecture. So there is no need to install any client software on nodes in order to manage them.

**INVENTORY**

The Inventory is a description of the nodes that can be accessed by Ansible. By default, the Inventory is described by a configuration file, whose default location is in./etc/ansible/hosts the configuration file lists either the IP address or hostname of each node that is accessible by Ansible. In addition, nodes can be assigned to groups.

**Syntax of inventory file:**

[webserver]

<private ip>

<private ip>

[testserver]

<private ip>

<private ip>

[all:children]

Webserver

Testserver

[all:vars]

ansible\_ssh\_user=<username>

ansible\_ssh\_pass=<password>

**PLAYBOOKS**

Playbook’s can be used to manage, configuration and deployments to remote machines.

Playbooks are designed to be a human-readable and are develop in a basic text language.

Playbooks are expressed in YAML Ani’t Markup Language (.yml) format.

Each playbook is composed of one or more play’s in a list.

The play’s main goal is to map a group of hosts to some well-defined roles, represented by things ansible call tasks.

--- 🡪indicated starting of playbook’s.

**-hosts:** one or more groups and host pattern.

**-tasks:** each playbook contains a list of tasks. Tasks are executing in order, one at a time, against all machine matched by the host pattern.

* The goal of each task is to execute module with very specific arguments, variables.
* Every task should have a name which is included in the output from running playbooks.

**-notify:** actions are triggered at the end of each block of task in a play, and will only be triggered once even if notified by multiple different tasks.

## **-handlers:** Running Operations on Change Handlers are lists of tasks, not really any different from regular tasks, that are referenced by a globally unique name, and are notified by notifies. If nothing notifies a handler, it will not run. Regardless of how many tasks notify a handler, it will run only once, after all the tasks complete in a particular play.

**Example playbook:**

$ vi playbook.yml

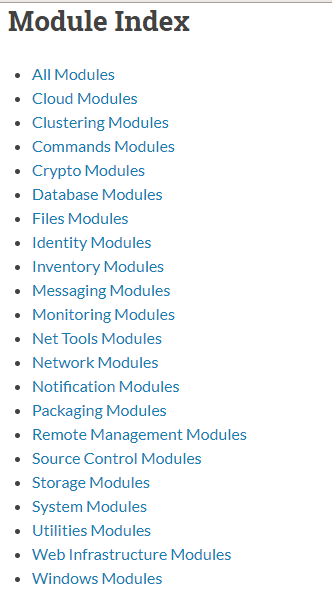
## 

## **Executing A Playbook**

$ansible-playbook playbook.yml

**MODULES**

There are over 450 modules provided by Ansible to automate every part of the environment. Modules are like plugins that do the actual work in Ansible, they are what gets executed in each playbook task. But you can also run a single one using the ‘ansible’ command. Each module is mostly standalone and can be written in a standard scripting language (such as Python, Perl, Ruby, Bash, etc.).



**Example of module:**

ansible webservers -m service -a "name=httpd state=started"

ansible webservers -m ping

**AD-HOC COMMAND**

An ad-hoc command is something that you might type in to do something really quick, but don’t want to save for later

**Ensure a service is started on all webservers:**

$ ansible webservers -m service -a "name=httpd state=started"

**Alternatively, restart a service on all webservers:**

$ ansible webservers -m service -a "name=httpd state=restarted"

**Ensure a service is stopped:**

$ ansible webservers -m service -a "name=httpd state=stopped“

**File transfer**

$ ansible webserver -m copy -a "src=/etc/hosts dest=/tmp/hosts"

**ROLES**

Roles are ways of automatically loading certain vars files, tasks, and handlers based on a known file structure. Grouping content by roles also allows easy sharing of roles with other users.

**Creating a role:**

* + $ ansible-galaxy install bennojoy.ngix
  + $ ansible-galaxy init <role name> 🡪 initializing a role.
  + $ ls 🡪 check role is created or not.
  + $ cd <role name> 🡪 change to created role galaxy

**Execution**

$ Vi Sampleplaybook1.yml

---

- host: web

become: yes

roles:

- role: <role name> 🡪

Execution a role-playbook:

$ ansible-playbook sampleplaybook1.yml