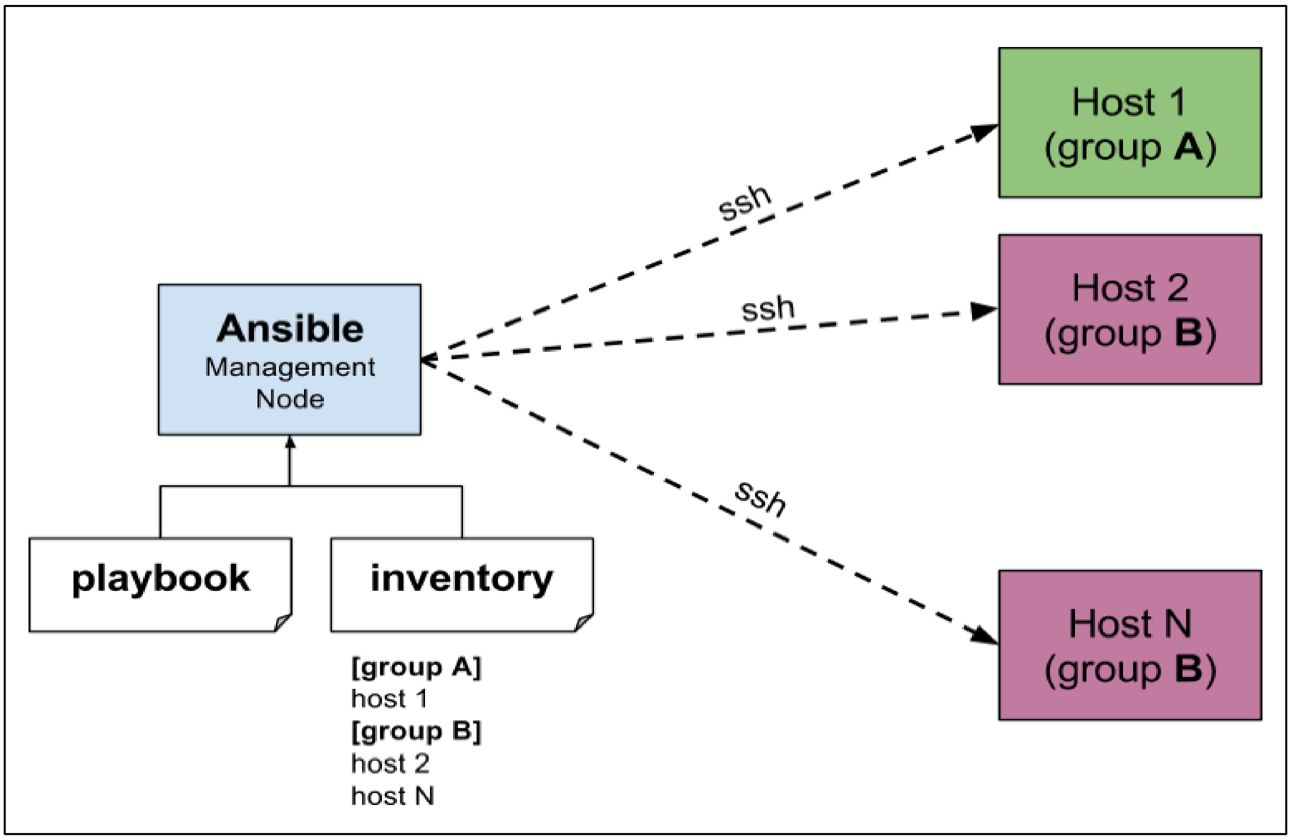
Ansible

- Ansible is the agentless configuration management tool which can automate the configuration of all the system. Ansible is written on python language and it works on push mechanism.

How Ansible Works:

Ansible works by connecting to nodes and pushing out small programs called "ansible module" to them.



```
vim /etc/ansible/hosts
[nodes:children]
groupA
groupB
```

Ansible Installation and Configuration:

- yum install ansible # To install ansible
- ansible --version   # To check version
- ansible-doc -l   # To list module
- ansible-doc copy   # To see details of copy modules.

Ansible

```
install=present
uninstall=absent
update=latest
start=started
stop=stopped
reload=reloaded
restart=restarted
```

-
  ```
  vim  /etc/ansible/ansible.cfg
  inventory = /etc/ansible/hosts
  host_key_checking = false
  ```

```
:wq
```

-
  ```
  vim /etc/ansible/hosts
  master ansible_host=master.hp.com
  node1 ansible_host=node1.hp.com
  node2 ansible_host=node2.hp.com
  node3 ansible_host=node3.hp.com
  node4 ansible_host=node4.hp.com

  [devops]   # Create a group of the nodes.
  master
  node1
  node2
  [nodesA]
  node3
  node4

  [nodes:children] # Merge two group in a another group.
  devops
  nodesA

  :wq
  ```

-
  ```
  vim /etc/ssh/sshd_config
  PermitRootLogin yes
  PasswordAuthentication yes
  : wq!

  systemctl restart sshd
  ```
  - Should be run on all hosts.

- Then need to setup ssh-key-based authentication for nodes and ansible server.

- ansible devops --list-host # To list all host of devops group
- ansible all --list-host   # List all host of all group
- ansible devops[1:2] -m ping
  0 = 1st node
  -1 = last node
  1 = 2nd node
  [1:2] = node1 and node2
- ansible devops:nodesA --list-hosts   # List the host of both nodes.

- **Inventory:** The ansible inventory file define the hosts and groups of hosts.

Format: -

alias   ansible_host=<FQDN> or < IP Address>

ansible_connection=ssh/winrm/localhost

Note: ssh: Linux | winrm: window | localhost: localhost

Example: -

server ansible_host=server.hp.com ansible_connection=ssh ansible_user=root ansible_ssh_pass=PS

server1 ansible_host=client.hp.com ansible_connection=winrm ansible_user=administrator ansible_password=CP

Config File: - /etc/ansible/hosts

Note: We have to uncomment “inventory = /etc/ansible/hosts” line from ansible configuration file “/etc/ansible/ansible.cfg”.

- **Playbooks:** Playbooks are the files where we define YAML script for execution of task.

- **Ad-hoc Command:** Ad-hoc commands are command which can be run individually to perform quick functions.

These ad-hoc commands are not used for configuration management and deployment because these commands will run one time only. And Ad-hoc command has no idempotency whereas module & playbook has idempotency.

Note: Idempotency means change command will not run unless needed (Means it’s won’t overwrite if already exist) and ansible will bring the system back to a desired state regardless of the actual state.

Ansible

Commands:

- ansible devops -a "ls /root"   # List the /root files of devops hosts.
- ansible devops -ba "yum remove httpd -y"   # Remove httpd
- ansible devops -a "sudo yum install httpd -y"   # Install httpd
- ansible devops -a /sbin/reboot   # restart
- ansible devops -ba "touch /root/file1"   # Create Files.

Note: b = become (Act as Sudo)
devops: host group name from inventory.

- **Module:** Module is reusable, standalone pre-define script that ansible runs on local or remote machine to perform some specific tasks.

  Commands:

  - COPY: ansible devops -m copy -a "src=/etc/yum.conf dest=/etc/book"
  - Create: ansible devops -m file -a "dest=/path/user1/new mode=777 owner=user1 group=user1 state=directory"
  - Delete: ansible devops -m file -a "dest=/path/user1/new state=absent"
  - Install:
    ansible devops -b -m yum -a "pkg=httpd state=latest"
    ansible devops -b -m service -a "name=httpd state=started" --check # To check status of service.
    ansible devops -b -m service -a "name=httpd state=started enables=yes"
  - User Creation: ansible devops -b -m user -a "name=raj"

Setup Command:

- ansible devops -m setup # Setup will check on remote node service is available or not.

YAML

- YAML stand for yet another markup language file extension should be .yml.
- YAML works on key: value pair and there should be one space after "key:" .
- All YAML files have to begin with "- - -"and end with ". . ."
- YAML has three principles.
  1. String Format

If one key a one value then it's known as String format.

Key: Value

2. Listing/Array Format

If one key has multiple value then it will be Array Format.

```
key:
- Value1
- Value2
- Value3
```

3. Mapping Format

Here one parent key and also nested key and under the nested key there will be the value

```
  key:
    key1: value

  ram:
        phone: 1234
        address: ABC
        skills:
        - python
        - java
```

Some Point Related to YAML.

To store diff property of an object we use dictionary like a car is object and it has diff - diff property model, colour, brand etc.

```
# key pair value
fruit: apple
meat: chicken
liquid: water

# A list of fruit and vegetables
fruits:     #list
-    mango # element of an array
-    apple
-    banana
vegetables:
-    carrot
-    tomato
```

Ansible

```
# Dictionary
fruits:
-    mango: # Nutrition info of two food.
          calories: 105  # These values are dictionary
          fat: 0.4g
          carbs: 27g
-    apple:
          calories: 105
          fat: 0.3g
          carbs: 16g
```

## Playbook

Playbook in ansible is written in YAML format and divided into many sections like,

- Target Section: Define the host against which playbooks task has to be executed.
- Variable Section: Define Variables.
- Task Section: List of all modules that we need to run in an order.

# Playbook to install httpd

Variables

Ansible uses variables which are defined previously to enable more flexibility in playbooks and roles. They can be used to loop through a set of given values, access various information like the host name of a system and replace certain strings in templates with specific values.

EX: Refer "Variables_demo" playbook.

Handlers Section

A handler is exactly the same as a task, but it will run when called by another task. Means it will depend on master task to be run successfully to run child task.

or

Handlers are just like regular tasks in an ansible playbook, but are run only if the task contains a notify directive and also indicates that it changed something.

Note: If we installing any package like httpd and also perform start the service so start will work only if httpd installation is completed.

Ex: Refer "Handler-Demo" playbook.

Ansible

Loops

Sometimes we want to repeat a task multiple time, In computer program this is called loops. common ansible loops including changing ownership on several files and/or directories with the file module, creating multiple users with the user module, and repeating a pooling step until certain result is reached.

Ex: Refer “Loops-Demo” playbook.

Conditions

Whenever we have diff-diff scenario, we have to use conditions according to the scenario.
like we want to install “apache2” package in Debian host and httpd on “RedHat” host.

When Statement:
sometime we want to skip a particular command on a particular node, we can use when statement.

Ex: Refer “Conditions-Demo” playbook.

Vault

Ansible use “AES256” technique to keep sensitive data such as playbook password or key in encrypted format, rather than a plain text in playbooks.

- Creating a new encrypted playbook.
  - ➢ ansible-vault create vault.yml

- Edit the encrypted playbook.
  - ➢ ansible-vault edit vault.yml

- To change the password.
  - ➢ ansible-vault rekey vault.yml

- To encrypt an existing playbook.
  - ➢ ansible-vault encrypt target.yml

- To decrypt an encrypt playbook.
  - ➢ ansible-vault decrypt target.yml