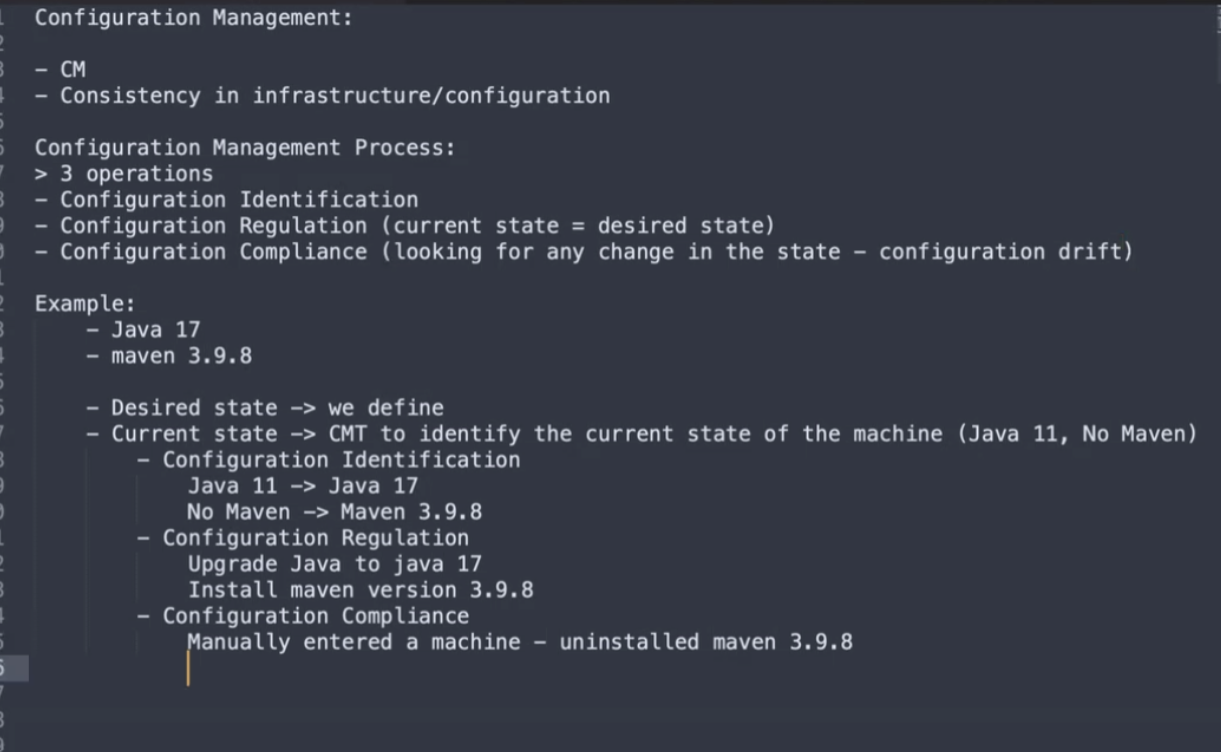
Configuration Management with Ansible and Terraform

Ansible is an open-source automation tool that simplifies IT tasks like configuration management, application deployment, and orchestration by using human-readable YAML files called playbooks.



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generatedTerraform is more a provisioning tool even if it can handle some aspects of configuration management (by provisioning and managing infrastructure resources).

Infrastructure as Code (IaC) is the practice of managing and provisioning computing infrastructure through machine-readable configuration files, rather than through physical hardware configuration or interactive configuration tools.

A screenshot of a computer program

Description automatically generatedA computer screen with white text

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

How to SSH to a machine: ssh username@ip\_hostname

Default port for SSH: 22

How to SSH to a particular machine: ssh username@ip\_hostname -p 42006

SSH

Into the same machine:

$ ssh localhost

$ ssh localhost -p 42006

# exit

Ansible: Passwordless SSH

Create a SSH keypair

[A]$ ssh-keygen

- Press Enter

- Overwrite? Press y

- Passphrase: Press Enter

- Confirm Passphrase: Press Enter

Copy the public key from Machine A

[A]$ cat ~/.ssh/id\_rsa.pub

Paste the public key to ~/.ssh/authorized\_keys on Machine B

[A & B - are the same machine]

[B]$ nano ~/.ssh/authorized\_keys

Paste the public key you have copied from Machine A

$ ssh localhost -p 42006

# exit

Ansible Verification:

$ ansible  --version

Ansible Inventory file:

$ sudo nano /etc/ansible/hosts

[myservers]

localhost

$ ansible -m ping localhost

Ansible Inventory file:

$ sudo nano /etc/ansible/hosts

[myservers]

localhost:42006

$ ansible -m ping localhost

$ ansible -m ping myservers

$ ansible all --list-hosts

$ ansible-inventory --graph

<https://docs.ansible.com/ansible/2.9/modules/list_of_all_modules.html>

Module help documentation:

$ ansible-doc

$ ansible-doc ping

Shell Module:

$ ansible all -m shell -a "hostname"

 ansible all -m shell -a "pwd"

$ ansible all -a "hostname"

$ ansible all -a "pwd"

Gather facts:

$ ansible all -m setup

$ ansible all -m setup -a 'filter=ansible\_hostname'

$ ansible all -m setup -a 'filter=ansible\_user\_\*'

$ ansible all -a uptime

$ ansible all -a "free -m"

$ ansible all -m apt -a "name=tree state=absent"

[Privilege escalation]

$ ansible all -m apt -a "name=tree state=absent" --become

[IF tree not installed]

$ ansible all -m apt -a "name=tree state=present"

[Privilege escalation]

$ ansible all -m apt -a "name=tree state=present" --become

$ ansible all -m file -a 'dest=/root/sample.txt state=touch mode=600 owner=root group=root' --become

$ ansible all -a "ls /root/sample.txt" --become

$ ansible all -m copy -a 'content="MY demo content" dest=/root/my\_demo.txt' --become

$ ansible all -a "cat /root/my\_demo.txt" --become

8/24/2024

Inventory file:

$ cat /etc/ansible/hosts

[myservers]

localhost:42006

Ansible Modules:

$ ansible all -m debug -a "msg=HelloWorld!"

Create, Validate & Parse YAML - using Python

$ nano yaml\_demo.py

import yaml

# Create a YAML structure & file

def create\_yaml():

data = {

'name': 'Abhi',

'age': '20',

'city': 'Mumbai'

}

with open('data.yaml', 'w') as file:

yaml.dump(data, file)

# Validate the YAML file

def validate\_yaml():

try:

with open('data.yaml', 'r') as file:

data = yaml.safe\_load(file)

print("Valid YAML file.")

return data

except yaml.YAMLError as exc:

print("YAML file error: ", exc)

return None

# Parse the YAML file

def parse\_yaml(data):

if data:

print("Name: ", data['name'])

print("Age: ", data['age'])

print("Location: ", data['city'])

if \_\_name\_\_ == "\_\_main\_\_":

create\_yaml()

my\_data = validate\_yaml()

parse\_yaml(my\_data)

Run the python file:

$ sudo python yaml\_demo.py

Configuring Apache Webserver using Ansible:

Check the status of apache webserver:

$ ansible all -m shell -a "service apache2 status"

$ nano apache2.yaml

---

- hosts: all

  become: true

  tasks:

    - name: Install apache2

      apt: name=apache2 update\_cache=yes state=latest

    - name: Enable mod\_rewrite

      apache2\_module: name=rewrite state=present

      notify:

        - restart apache2

  handlers:

    - name: restart apache2

      service: name=apache2 state=restarted

Execute the playbook:

$ ansible-playbook apache2.yaml

Check the status of apache webserver:

$ ansible all -m shell -a "service apache2 status"

Anchors & Aliases in YAML:

$ nano users.yml

---

users:

  - &default\_address

    address:

      street: "ABC Main St"

      city: "Sometown"

      state: "CA"

      zip: "12345"

  - name: "Alice"

    <<: \*default\_address

  - name: "Bob"

    <<: \*default\_address

  - name: "Charlie"

    <<: \*default\_address

$ nano print\_users.yml

---

- name: Print the users with default address

  hosts: all

  gather\_facts: no

  vars\_files:

    - users.yml

  tasks:

    - name: Print user details

      debug:

        msg: "User: {{ item.name }} lives at {{ item.address.street }}, {{ item.address.city }}, {{ item.address.state }}, {{ item.address.zip }}"

      loop: "{{ users }}"

      when: item.name is defined

$ ansible-playbook print\_users.yml --syntax-check

$ ansible-playbook print\_users.yml

Global Variable in Ansible:

$ mkdir ansible\_demo\_01

$ cd ansible\_demo\_01

$ mkdir group\_vars

$ nano group\_vars/all.yml

---

greeting: "Hello, World!"

custom\_number: 56

$ nano demo\_playbook.yml

---

- name: Demo playbook for group vars testing

  hosts: all

  gather\_facts: no

  tasks:

    - name: Print the greeting message

      debug:

        msg: "{{ greeting }}"

    - name: Print the custom number

      debug:

        msg: "{{ custom\_number }}"

$ ansible-playbook demo\_playbook.yml

YAML for Data serialization using Python:

$ cd

$ nano serialize.py

import yaml

#Create a Python dictionary

data={

'title': 'YAML serialization example',

'version': 1.0,

'items': ['item1', 'item2', 'item3']

}

#Serialize the dictionary to a YAML file

with open('serial\_output.yaml', 'w') as file:

yaml.dump(data, file, default\_flow\_style=False)

print("Data serialization complete - serial\_output.yaml")

$ python serialize.py

Install NodeJS using Ansible Playbook execution:

$ nano node.yaml

---

- name: Install nodejs

  hosts: all

  gather\_facts: yes

  become: true

  tasks:

    - name: Add apt key for node installation source

      apt\_key: url=https://deb.nodesource.com/gpgkey/nodesource.gpg.key

    - name: Add the node repo

      apt\_repository:

        repo: 'deb https://deb.nodesource.com/node\_0.10 {{ ansible\_distribution\_release }} main'

        update\_cache: no

    - name: install nodejs

      apt: name=nodejs

$ ansible-playbook node.yaml

$ ansible all -m shell -a "node --version"

$ ansible-playbook node.yaml --verbose

$ ansible-playbook node.yaml --check

$ ansible-playbook node.yaml --syntax-check

Variables in Ansible:

$ nano basic.yaml

---

- hosts: all

  vars:

    salutations: Hello everyone!!!

  tasks:

    - name: Using Ansible variable

      debug:

        msg: "{{ salutations }}"

$ ansible-playbook basic.yaml

Loops:

$ nano loops.yaml

---

- name: My loop example

  hosts: all

  vars:

    my\_list\_var:

      - my\_list\_01

      - my\_list\_02

    my\_dict\_var:

      name: "Abhi"

      type: "demo"

  tasks:

    - name: loop with direct list value

      debug:

        msg: "{{ item }}"

      loop:

        - one

        - two

    - name: loop with list var

      debug:

        msg: "{{ item }}"

      loop: "{{ my\_list\_var }}"

    - name: with\_items for list var

      debug:

        msg: "{{ item }}"

      with\_items: "{{ my\_list\_var }}"

    - name: with\_dict for dict var

      debug:

        msg: "New key: {{ item.key }}, New value: {{ item.value }} "

      with\_dict: "{{ my\_dict\_var }}"

    - name: with\_items for list var

      debug:

        msg: "{{ item.0 }} - {{ item.1 }}"

      with\_indexed\_items:

        - one

        - two

$ ansible-playbook loops.yaml

Does that mean that I successfully created the inventory file web\_app to define the remote server?