**SETUP AND CONFIGURATION OF ANSIBLE**

STEPS: (docs.ansible.com)

* Open 4 EC2 instance in your AWS(Provision using the codes for each) (1 for Ansible, 2 for Webservers & 1 for Database)
* **Ansible –**

(Ubuntu, Key = Name, Value = Control machine)

Security Group (Name = Control-SG, SSH/TCP/22/myip)

Keypair (Name = Control-key)

* 2 Webservers –

(Centos7, Instances = 2, Key = Name, Value = Vprofile-web01 and 02 differently)

Security Group (Name = vprofile-web-sg, SSH/TCP/22/myip, SSH/TCP/22/control-sg)

Keypair (Name = vprofile-key)

* Database –

(Centos7, Key = Name, Value = Vprofile-db01)

Security Group (Name = vprofile-db-sg, SSH/TCP/22/myip, SSH/TCP/22/control-sg)

Keypair (Name = use same key for webserver)

* Install Ansible:(Provisioning Codes – Use Ubuntu 18)
* #!/bin/bash
* sudo apt update
* sudo apt install software-properties-common -y
* sudo apt-add-repository –yes –update ppa:ansible/ansible
* sudo apt install ansible –y
* SSH to the Ansible (Control Machine) using GitBash

ssh –i Download/control-key.pem ubuntu@IP

* ansible --version (To check Ansible version)
* mkdir vprofile
* cd vprofile/
* mkdir exercise1 (Exercise refers to the no of projects done)
* cd exercise1
* **Checking the inventory & Ping Module (Inventory file contains info of all the nodes or hosts)**
* sudo ls /etc/ansible/hosts (this is the default inventory file)
* vim inventory (To create our project inventory file)
* web01 ansible\_host=(private Ip of web01) ansible\_user=centos ansible\_ssh\_private\_key\_file= (follow the steps below)
* :wq
* logout
* cat Downloads/vprofile-key.pem (copy the key)
* ssh –i Download/control-key.pem ubuntu@IP of the control machine
* cd vprofile/exercise1/
* vim vprofile-key.pem (paste the copied key here)
* :wq
* ls –l
* chmod 400 vprofile-key.pem
* vim inventory
* web01 ansible\_host=(private Ip of web01) ansible\_user=centos ansible\_ssh\_private\_key\_file= vprofile-key.pem
* web02 ansible\_host=(private Ip of web02) ansible\_user=centos ansible\_ssh\_private\_key\_file= vprofile-key.pem
* db01 ansible\_host=(private Ip of db01) ansible\_user=centos ansible\_ssh\_private\_key\_file= vprofile-key.pem
* :wq
* sudo vim /etc/ansible/ansible.cfg
* scroll down to line 71, uncomment the: host\_key\_checking = False
* :wq
* Test the module on your targets ----
* ansible –i inventory -m ping web01 (it should show a green message and success)
* ansible –i inventory -m ping web02
* ansible –i inventory -m ping db01
* To connect to more than one node at a time: (you create groups)
* vim inventory
* web01 ansible\_host=(private Ip of web01) ansible\_user=centos ansible\_ssh\_private\_key\_file= vprofile-key.pem
* web02 ansible\_host=(private Ip of web02) ansible\_user=centos ansible\_ssh\_private\_key\_file= vprofile-key.pem
* db01 ansible\_host=(private Ip of db01) ansible\_user=centos ansible\_ssh\_private\_key\_file= vprofile-key.pem
* [websrvgrp]

web01

web01

* [dbsrvgrp]

db01

* [dc\_ohio:children]

websrvgrp

dbsrvgrp

:wq

* Test the module on your targets in groups ----
* ansible –i inventory -m ping websrvgrp (it should show a green message and success)
* ansible –i inventory -m ping dbsrvgrp
* ansible –i inventory -m ping dc\_ohio:children
* ansible –i inventory -m ping all (To run on all nodes or hosts)

OR:

ansible –i inventory -m ping ‘\*’

* To define the variables at the Global level
* vim inventory
* web01 ansible\_host=(private Ip of web01)
* web02 ansible\_host=(private Ip of web02)
* db01 ansible\_host=(private Ip of db01)
* [websrvgrp]

web01

web01

* [dbsrvgrp]

db01

* [dc\_ohio:children]

websrvgrp

dbsrvgrp

* [dc\_ohio:vars]

ansible\_user=centos

ansible\_ssh\_private\_key\_file= vprofile-key.pem

:wq

* Test the module on your targets in groups ----
* ansible –i inventory -m ping websrvgrp (it should show a green message and success)
* ansible –i inventory -m ping dbsrvgrp
* ansible –i inventory -m ping dc\_ohio:children
* ansible –i inventory -m ping all (To run on all nodes or hosts)

OR:

ansible –i inventory -m ping ‘\*’

* **Ad-Hoc Commands: (Installing a package in a server .e.g. web01)**
* cd vprofile/
* cp –r exercise1/ exercise2
* cd exercise2/
* ansible –i inventory –m ping all (to test the connection)
* ansible –i inventory –m yum –a “name=httpd state=present” web01 –become (to install hppd)
* IDEMPOTENT: These means if the system is in the same state it wouldn’t apply the state again .E.g. “Changed”: false ---- means nothing changed
* ansible –i inventory –m service –a “name=httpd state=started enabled=yes” web01 –become (to start hppd)
* vim index.html

This is deployed by Ansible.

:wq

* ansible –i inventory –m copy –a “src=index.html dest=/var/www/html/index.html” web01 –become (To copy files to a directory)
* PLAYBOOKS & MODULES: Playbooks are written in YAML format
* login to the control machine (Ansible EC2 created above) using GitBash

ssh –i Downloads/control-key.pem ubuntu@public IP of the machine

* mkdir vprofile (if the directory doesn’t exist. But it was done above)
* cd vprofile/
* ls
* cp –r exercise2 exercise3
* cd exercise3/
* ansible –i inventory –m ping all (to test connection)
* vim web\_db.yaml

- - -

- name: Setup Webserver

hosts: websrvgrp

become: yes

tasks:

* name: Install Apache httpd

yum:

name: httpd

state: present

* name: Start & Enable HTTPD

service:

name: httpd

state: started

enabled: yes

- name: Setup DBserver

hosts: dbsrvgrp

become: yes

tasks:

- name: Install MySQL server

yum:

name: mariadb-server

state: present

- name: Start & Enable mariadb-server

service:

name: mariadb

state: started

enabled: yes

:wq

* ansible-playbook –i inventory web\_db.yaml - -syntax-check (If it results the name of the playbook it means the syntax is good)
* ansible-playbook –i inventory –m yum –a “name=httpd state=absent” - -become web01 (This is to remove httpd server if already installed, so as the next command will be successful)
* ansible-playbook –i inventory web\_db.yaml (to execute the playbook)
* vim index.html

This is deployed by Ansible playbook

:wq

* vim web\_db.yaml (To copy index.html into the webserver)

- - -

- name: Setup Webserver

hosts: websrvgrp

become: yes

tasks:

* name: Install Apache httpd

yum:

name: httpd

state: present

* name: Start & Enable HTTPD

service:

name: httpd

state: started

enabled: yes

* name: Deploy web file

copy:

src: index.html

dest: /var/www/html/index.html

- name: Setup DBserver

hosts: dbsrvgrp

become: yes

tasks:

- name: Install MySQL server

yum:

name: mariadb-server

state: present

- name: Start & Enable mariadb-server

service:

name: mariadb

state: started

enabled: yes

:wq

* ansible-playbook –i inventory web\_db.yaml (To execute, to test if the website opens)
* **To view all Ansible Modules:**
* ansible-doc –l (shows a list of modules)
* ansible-doc yum
* ansible-playbook –i inventory web\_db.yaml -C (To show a preview of what the outcome of your codes is. If it executes successfully doesn’t guarantee it success in its actual use)
* **How to find the right Module, use & Troubleshoot**
* cd vprofile/
* ls –ltr
* cp –r exercise3 exercise4
* cd exercise4/
* we want to add SQL database and add a user and give privilege to the user in a Database
* Go to google and type (ansible documentation modules index)
* click on the official doc site
* click on Database Modules in the list on the right pane
* scroll down to MySQL section and click on:

([mysql\_db – Add or remove MySQL databases from a remote host](https://docs.ansible.com/ansible/2.9/modules/mysql_db_module.html" \l "mysql-db-module))

([mysql\_user – Adds or removes a user from a MySQL database](https://docs.ansible.com/ansible/2.9/modules/mysql_user_module.html" \l "mysql-user-module))

* check below for use case in the Yaml file
* mv web\_db.yaml db.yaml
* cat inventory (to view the IP and Host info)
* ssh –i vprofile-key.pem centos@Ip of db01 (we need to install the dependency in db01)
* sudo –i
* yum search python | grep –i mysql
* copy the name of the dependency result shown (MySQL-python) (we would install it through the playbook)
* logout
* logout
* SSH to the control machine (Ansible)

ssh –i Downloads/control-key.pem ubuntu@public IP of the machine

* cd /vprofile/exercise4
* vim db.yaml (we want to add a new database and add a user and give privilege to the user)

- - -

- name: Setup DBserver

hosts: dbsrvgrp

become: yes

tasks:

- name: Install MySQL server

yum:

name: mariadb-server

state: present

- name: Install Python MySQL

yum:

name: MySQL-python

state: present

- name: Start & Enable mariadb-server

service:

name: mariadb

state: started

enabled: yes

**-** name**:** Create a new database with name 'accounts'

mysql\_db**:**

name**:** accounts

state**:** present

**-** name**:** Create database user with name 'admin'

mysql\_user**:**

name**:** admin

password**:** 12345

priv**:** '\*.\*:ALL'

state**:** present

:wq

* ansible-playbook –i inventory db.yaml (To test the module)
* **Ansible Configurations (Changing Default settings .E.g. SSH settings etc)**
* The Order of Ansible Config:

1. ANSIBLE\_CONFIG(environment variable if set) (This is at the system level)
2. ansible.cfg(in the current directory) (Recommended level)
3. ~/.ansible.cfg(in the home directory)
4. /etc/ansible/ansible.cfg (This is at the system level)

STEPS:

* Login to the control machine
* sudo –i
* cd /etc/ansible/
* ls
* cd
* vim /etc/ansible/ansible.cfg

if you don’t see the settings, copy the command (ansible-config init - - disabled > ansible.cfg

:q!

* cd /etc/ansible/
* mv ansible.cfg ansible.cfg\_backup (take backup of the existing configuration file)
* ansible-config init - - disabled > ansible.cfg
* ls (you should see a new config file)
* vim ansible.cfg

:se nu (shows the line nos)

# and ; are used to comment a line

:q!

* vim /etc/ansible/ansible.cfg (this is the global config file)

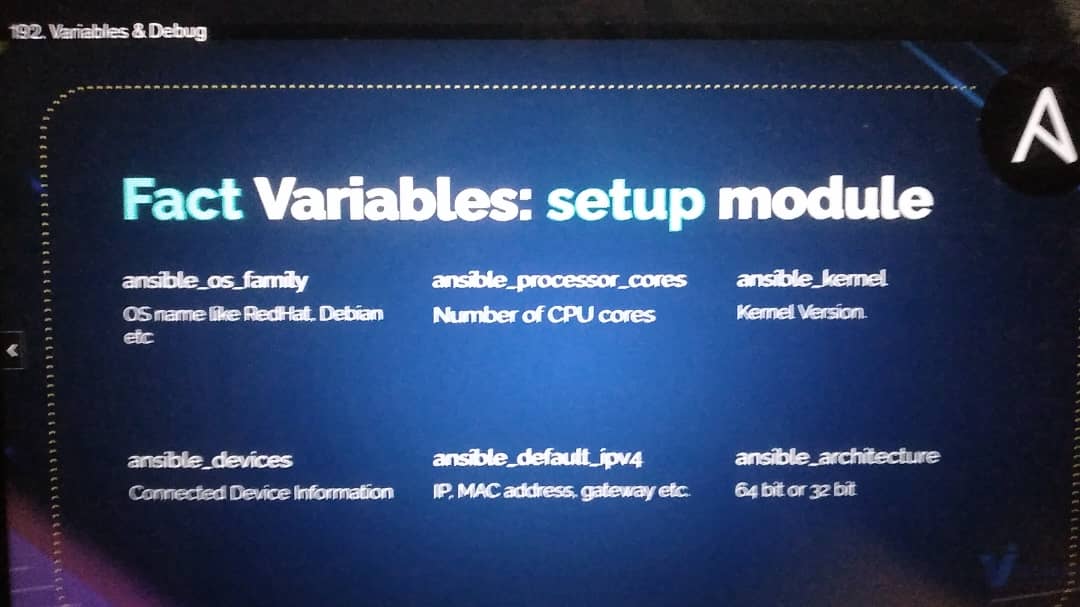
(doc.ansible.com – check configuration for explanation of each settings)

:q!

* logout (to leave root user)
* cd vprofile/
* ls –ltr
* Creating our own configuration file
* cp –r exercise4 exercise5
* cd exercise5/
* ls
* vim ansible.cfg
  + - [defaults]
    - host­\_key\_checking = False
    - inventory = ./inventory
    - forks = 5
    - log\_path = /var/log/ansible.log
    - [privilege escalation]
    - become=True
    - become\_method=sudo
    - #become\_user=root
    - become\_ask\_pass=False

:wq

* + - sudo touch /var/log/ansible.log
    - sudo chown ubuntu.ubuntu /var/log/ansible.log
    - ansible-playbook db.yaml (To execute and Test the configuration)
    - cat /var/log/ansible.log
    - ansible-playbook db.yaml –v (-v means level of robustity. It helps in troubleshooting. The no of –v can be increased to view more results)
    - cat /var/log/ansible.log (More detailed info should show)
* **Variables & Debug**



STEPS TO USE VARIABLES:

* cp –r exercise5 exercise6
* cd exercise6/
* vim db.yaml

- - -

- name: Setup DBserver

hosts: dbsrvgrp

become: yes

vars:

dbname: groups

dbuser: devops

dbpass: admin123

tasks:

- debug:

var: dbname

- debug:

msg: “The Value of dbuser is {{dbuser}}”

tasks:

- name: Install MySQL server

yum:

name: mariadb-server

state: present

- name: Install Python MySQL

yum:

name: MySQL-python

state: present

- name: Start & Enable mariadb-server

service:

name: mariadb

state: started

enabled: yes

**-** name**:** Create a new database with name 'accounts'

mysql\_db**:**

name**:** “{{dbname}}”

state**:** present

**-** name**:** Create database user with name 'admin'

mysql\_user**:**

name**:** “{{dbuser}}”

password**:** “{{dbpass}}”

priv**:** '\*.\*:ALL'

state**:** present

:wq

* ansible-playbook –i inventory db.yaml (To test the module)

**Group and Host Variables: (How to hide information)**

* Here the password and sensitive infos are removed from the YAML file and placed in a separate script to hide it.
* cp –r exercise6 exercise7
* cd exercise7/
* mkdir group\_vars
* vim group\_vars/all
* # Variables is accessible by All the Hosts from the Inventory file
* dbuser: testadmin
* dbpass: admin123
* dbname: redhat
* :wq
* vim db.yaml

- - -

- name: Setup DBserver

hosts: dbsrvgrp

become: yes

# vars:

# dbname: groups

# dbuser: devops

# dbpass: admin123

tasks:

- debug:

var: dbname

- debug:

msg: “The Value of dbuser is {{dbuser}}”

tasks:

- name: Install MySQL server

yum:

name: mariadb-server

state: present

- name: Install Python MySQL

yum:

name: MySQL-python

state: present

- name: Start & Enable mariadb-server

service:

name: mariadb

state: started

enabled: yes

**-** name**:** Create a new database with name 'accounts'

mysql\_db**:**

name**:** “{{dbname}}”

state**:** present

**-** name**:** Create database user with name 'admin'

mysql\_user**:**

name**:** “{{dbuser}}”

password**:** “{{dbpass}}”

priv**:** '\*.\*:ALL'

state**:** present

:wq

* ansible-playbook –i inventory db.yaml (To test the script all inside group\_vars( vim group\_vars/all))

**Script to execute on all host (To create a User)**

* cp –r exercise7 exercise8
* cd exercise8/
* rm –rf db.yaml index.html group\_vars/all
* vim vars\_precedence.yaml

- name: Understanding precedence of variables

hosts: all

become: Yes

vars:

USRNM: playuser

COMM: variable from playbook

tasks:

- name: create a user

user:

name: “{{USRNM}}”

comment: “{{COMM}}”

register: USROUT

- debug:

var: USROUT.name

- debug:

var: USROUT.comment

:wq

ansible-playbook vars\_precedence.yaml

**Script to execute on all host (To create a User) Here, the variable in the YAML file is of higher precedence, it reads the variable in yaml first:**

* mkdir group\_vars
* vim group\_vars/all

USRNM: globaluser

COMM: value from group\_vars/all file

:wq

* vim vars\_precedence.yaml

- name: Understanding precedence of variables

hosts: all

become: Yes

vars:

USRNM: playuser

COMM: variable from playbook

tasks:

- name: create a user

user:

name: “{{USRNM}}”

comment: “{{COMM}}”

register: USROUT

- debug:

var: USROUT.name

- debug:

var: USROUT.comment

:wq

ansible-playbook vars\_precedence.yaml

* **ANSIBLE FOR AWS (CREATE KEY PAIRS, CREATE AND START AN EC2 INSTANCE)**

**(CHECK DOCUMENTATION :doc.ansible.com)**

* Login to your AWS account
* create an EC2 instance for Ansible as explained in the first page of these doc
* create an IAMUSER
* username = ansibleadmin,
* programmatic access
* attach existing policies directly (select: administratorAccess)
* save the key
* Be very careful with the access key from Hackers
* open GitBash, ssh to the Ansible EC2 machine
* vim .bashrc (scroll to the end, last line below fi closure)

export AWS\_ACCESS\_KEY\_ID=’from the csv file saved from IAMUSER’

export AWS\_ SECRET\_ACCESS\_ KEY=’from the csv file saved from IAMUSER’

* source .bashrc (Authentication will be successful )
* mkdir aws
* cd aws/
* sudo apt install python-pip -y
* pip install boto3
* pip install boto
* vim test-aws.yaml

- hosts: localhost

gather\_facts: False

tasks:

- name: Create key pair

ec2\_key:

name: sample

region: us-east-1

register: keyout (stores the output and passes it on to a variable )

- name: print key

debug:

var: keyout

- name: save key

copy:

content: “{{keyout.key.private\_key}}”

dest: ./sample.pem

when: keyout.changed

- ec2

key\_name: sample

instance\_type: t2 micro

image: (go to a region, click on create Ec2 and copy AMI ID of the OS system)

region: us-east-1

wait: yes

group: webserver

exact\_count: 1

instance\_tags:

Name: db01

db: postgres

count\_tag:

Name: db01

db: postgres

:wq

* ansible-playbook test-aws.yaml
* cat sample.pem (to get the saved private key)

complete training 193 – 199