Create EC2 instance : cloud connectivity with Ansible :-

→ First creat a connectivity b/w cloud and control machine.

→ For that we have to go a service from cloud i.e IAM.

→ IAM : 'Indentity Access Managment' service.

→ search IAM and go to IAM in cloud.

creat users ⇒ Users

→ Add users.

→ user name :- ansible.

→ Provide access

→ Require pass rest ☐ 'disable'

→ Permissions.

→ Attach existing policies directly

→ Amazon EC2 full access. ☑

Administrator access.

→ creat user

→ Download.csr.

Now, you will get 'Access key' and secret key. past them in control machine as root_user

export AWS_ACCESS_KEY_ID = "Access key"

export AWS_SECRET_ACCESS_KEY = "secret key"

Now,

Install python.

Yum install python-pip.

↓

→ Do it from root user.

pip install boto boto3

pip show boto

pip show boto3.

cd playbooks/ on ansible user

Vi ec2.yml.

→ Paste playbook from Pdf.

→ key_name : from instance (name of key) ✓

image : ami - of (Ubuntu) ✓

region :

group :

count :

wait :

assign_public_ip: Yes

vpc_subnet_id : ✓

search vpc

subnets

take the subnets available.

v = change

ansible-playbook ec2.yml

→ if error occured the export the "Access key" & "Secret key".

Note :- it is a session level process of connecting b/w

cloud and ansible.

Handlers:-

Playbook require No. of. time to restart this are going to execute at end of playbook and is only once

vi handlers.yml

→ Paste the code from pdf

→ download the source ie sample.war (copy the link)

wget (paste the link)

ansible-playbook handlers.yml

when condition:-

for placing the condition

vi when.yml

→ Paste the code from Pdf

***

ansible webservers -m setup

⇒ To know the details of the remote server ex:- IPaddress, os etc.,

ansible-playbook when.yml

Variables:- 'vars'

vi vars.yml

⇒ Paste the code from Pdf

⇒ To paste variable use two {{ }}

⇒ Create Source

touch index.html

echo "This is ansible vars DEMO" > index.html

ls

prod

→ index.html is created in /home/ansible/playbook

so move to /home/ansible

mv index.html /home/ansible/

ls

ansibe-playbooks vars.yml

Jinja2 Templati :-

Template are for Re-usability

~~vi apache-conf.j2~~

→ create a file

touch apache-conf.j2

→ on Remote server

find / -name httpd conf

cat /etc/httpd/conf/httpd.conf (copy this data)

vi apache-conf.j2 and save

vi apache-conf.j2 (to make templet)

replace with variable listen {{ Port_number }}.

vi templet.yml

→ Paste from Pdf.

Roles :- is a preceses of a playbook ex:- ~~Hosts~~
vars
Tasks
Handlers

Manual way of Role creation :-

Go to root user.

cd /etc/ansible

ls

cd roles/

ls → (roles empty)

mkdir apache

ls

cd apache/ → (create roles)

mkdir files handlers meta tasks vars

ls

cd tasks/

ls

Vi main.yml → (by default role will use main.yml file)

→ Paste the code from Pdf

cd ..

cd handlers/

~~vi~~ vi main.yml

→ Paste the code (handlers)

cd ..

cd vars/

ls

vi main.yml

→ Paste the vars code

cd ..

cd files/ (roles place in files folder).

ls

touch index.html

echo "This is Roles demo of Ansible" > index.html

cat index.html

ls

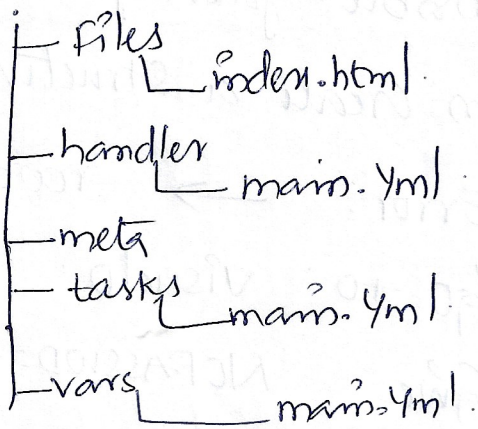
cd..

ls

Yum install tree

tree

→ we can see structure



Now, execute the role. for that we need a playbook

vi deploye.yml → execute command

→ Paste the code

Go to ansible

ansible-playbook deploy.yml

Su ansible

cd

ansible webservers -m ping.

cd /etc/ansible/

ls

cd roles/

ls

cd apache/

ls

→ ansible-galaxy init tomcat --offline.

We can create a structure ex:- Tomcat role.

→ Error. → root user then.

Go to visudo.

Give NOPASSWD=ALL

Sudo ansible-galaxy init tomcat --offline.

cd tomcat/.

ls cd tasks/

Sudo vi main.yml

→ Paste the code (tomcat).

cd.. cd handlers.

Sudo vi main.yml

→ Paste the code.

cd ..

ls

cd files/

→ download war.file

sudo wget (war.file path)

ls

cd..

ls

cd..

ls

cd..

ls

pwd

→ add this roles in deploy.yml

vi deploy.yml

→ Paste the ~~past~~ code.

ansible-playbook deploy.yml

→ we get roles from 'community' Pre-defined roles.

Google :- ansible galaxy (galaxy.ansible.com).

⇒ import (playbook calling another playbook) 8:50

playbook

cd playbooks/

ls

sudo vi user.yml

→ Paste the code from Pdf.

Here filecreat.yml is already present.

```
import_playbook : filecreate.yml
```

Here user.yml (playbook) is calling another playbook (filecreate.yml) using import_playbook

~~Included~~ import_tasks (one piece of role using another role)

cd /etc/ansible/roles/

ls

→ creating a role format.

sudo ansible-galaxy init foo --offline.

ls

cd foo/

ls

cd tasks/

ls

cat main.yml

→ nothing is present.

sudo vi main.yml

→ Paste the code - import_task = "/etc/ansible/roles/tomcat/tasks/main.yml"

cd .. /.. /..

→ Now, Go to 'deploy.yml' file

sudo vi deploy.yml

→ Paste the Role 'Foo'

ansible-playbook deploy.yml.

playbook to role

Include task (~~Role calling playbook~~).

Go to 'deploy.yml' file which is in cd /etc/ansible/

sudo vi deploy.yml

→ Add task:
- include: /etc/ansible/roles/foo/tasks/main.yml

ansible-playbook deploy.yml

Handling data :- Vaults.

cd

* ansible-vault create vault.yml. (To create the vault pass)

  v pass : ansible.

  pass : ansible.

  → Paste the code.

* ansible-vault view (file name).

⇒ to see the data, after we provide password.

* ansible-vault edit (file name)

⇒ to update the data

* ansible-vault rekey vault.yml

⇒ to reset password

* ansible-vault decrypt (file name).

⇒ To remove the password.

* ansible-vault encrypt vault.yml

⇒ To again create the password

Now, execute this data on 'RM' servers. for that

ansible --ask-vault-pass webservers -m copy -a

'src=vault.yml dest=/temp/secret_key'

ansible webservers -m command -a 'cat /tmp/secret_key'

Without prompting of vault password :-

Password

echo 'ansible' > .vault_pass

ls

ls -a

ansible --vault-password-file=.vault_pass -m 'src=vault.yml dest=/temp/secret_key mode=0600' webservers --become

Permanent creation

/etc/profile

profile.d create a file

**

cd /etc/ansible/

vi ansible.cfg

→ # vault_password_file = /path/to/vault_pass_file

vault_Password_file = /home/ansible/.vault_pass

mention the path of vault

ansible -m copy -a 'src=vault.yml dest=/temp/secret_key mode=0600' webservers --become

Dynamic inventory :- (For Testing Environment)

Create IAM user with EC2 full access.

git clone (path).

Yum install git -y

git clone (path).

ls

cd devops-1/

ls

cd ansible/

ls

cd inventory/

ls

cat ec2.ini

cat ec2.py.

IAM service

user

Add users

Run, export the keys secret & Access key.

chmod 755 ec2.py

./ec2.py --list.

→ This files will give the details of machines running on AWS

Now, to use this inventory file.

ansible -i /root/devops-1/ansible/inventory/ec2.py

--Private-key /home/ec2-user/ (private key name)

(tag-(machine name) -m ping

to get private-key

Take public-key of 'CM' and paste in 'Host name'

User name : ec2-user

~~A~~dvanced

Advanced

Auth

PPK file

login

and drag n drop

pem file

Putty gen

Run

load

~~Putty~~ All files

Pem file

open

Save

~~Save~~ Vamshi-devops

Save

cd /home/ec2-user/

ls

cp Scriptkaro cd

ansible -i /root/devops-1/ansible/inventory/ec2.py

--private-key /home/ec2-user/(Key name) (machine)

-m ping.

cd /etc/ansible/

vi ansible.cfg (with out pass inventory) changed one

⇒ inventory = /root/devops-1/ansible/inventory/ec2.py

inventory = /etc/ansible/hosts/ ~~changed~~

→ ansible --private-key /home/ec2-user/ramshi-devops.pm -u ec2-user (tag-name) -m ping.

vi ansible.cfg (without private key).

⇒ # private_key_file = /path/to/file

private_key_file = /home/ec2-user/ (pem key)

→ ansible -u ec2-user (tag-name) -m ping.

vi ansible.cfg (without user).

remote_user = root

remote_user = ec2-user.

ansible (tag-name) -m ping.

Sample Jenkins pipeline:-

→ Why should we go to Pipeline?

1) Faster Execution
2) Stage level Log
3) Parallel Execution of stages
4) Replay from Particular stage
5) Multi branch Pipeline.

Pipeline are of two types:-

i) declarative - supports groovy
ii) scripted (groovy)

i) Pipeline starts with keyword

→ Pipline {
agent → Properties like inputs & triggers & tools.
main task → stages { stage } & Build.

Post {
success:
failur:
} & Post-build

post can be build level (or) job level (like individual)

2) Scripted :- (keyword is Node)

Node {→

***

To Get the code :- (multi-use (or) example codes)

Pipeline-Syntax

Declerative = Pipeline (Keyword)

Scripted = Node (keyword).

Snippet Generator :-

Here, we can get code of ((Github) (maven) (Sonarqube) (Nexus))