**Ansible Assignment 3**

1. Could you construct a simple playbook to install Nginx on a server?

* with ansible most easiest way to implement is to install and configure a webserver or nginx
* login to management machine and check if webserver is accessible or not by using say curl. assume webserver ip is 192.168.10.10 ( this can vary as per configuration done in real time)
* suppose there is no webserver on machine so below should fail

curl http://192.168.10.10

* Now let us install nginx
* for that we need to have an inventory file that list of all servers you need to manage like eg. below is content in inventory file

|  |
| --- |
| [web]  192.168.56.11  [web:vars]  ansible\_python\_interpreter=/usr/bin/python3 |

* let's have a group called "web" (you can have any name) and for now lets only have single remote machine listed in that group. The ansible\_python\_interpreter=/usr/bin/python3 is there so Ansible will use Python 3 on the remote machine.
* create yaml file eg. examples/ansible/nginx\_install.yaml

|  |
| --- |
| ---  - hosts: all  tasks:  - name: ensure nginx is at the latest version  apt: name=nginx state=latest  - name: start nginx  service:  name: nginx  state: started |

* In above yaml we have one task and two names. names are just for human readable
* The first step apt: name=nginx state=latest tells the apt module that we want the latest version of the nginx package to be installed. The next step tells Linux to make sure the service called nginx is started

ansible-playbook -i inventory nginx\_install.yaml -u user1 -K

* Because the playbook requires sudo to run, we’re also including the -K argument to provide the remote user’s sudo password when prompted by Ansible:

OR

ansible-playbook -i inventory.cfg nginx\_install.yaml -b

* if you have set up passwordless sudo then we need not to pass -K we can use -b means ansible to become root on the remote server using sudo

check the output

now you can try curl http://192.168.10.10

We can also install and configure in another way like :

* Step 1: Let us first generate a public SSH key to connect to your host.

ssh-keygen

* Step 2: Now copy the public SSH key on your hosts eg.

ssh-copy-id -i root@IP\_of\_target\_machine

* Step 3: Now list the IP addresses of your hosts/nodes in your inventory.

vi /etc/ansible/hosts

add the server ip. in the inventory with group name let's say [test-server]

|  |
| --- |
| [test-server]  192.168.10.10 |

* Step 4: Now let's check if the connection has been established using ping

**ansible -m ping 'test-server'**

* Step 5: Now we create a playbook to install Nginx on the host machine.

create a file with yml extention vi nginx\_demo.yml

|  |
| --- |
| --  hosts: stagingwebservers  sudo: yes  vars:  - server\_port: 8080    tasks:  - name: install nginx  yum: pkg=nginx state=installed    - name: serve nginx config  template: src=../files/flask.conf dest=/etc/nginx/conf.d/  notify:  - restart nginx  handlers:  - name: restart nginx  service: name=nginx state=restarted |

Save the file exit

in above playbook first task is to get the required package for Nginx and then install it.

Ansible will check if the directory exists and create it if it’s not otherwise, it will do nothing.

Handlers are define to take a action only when notification of tasks or state changes.

In this playbook we have defined notify: restart Nginx handler which will restart Nginx once the files and templates are copied to hosts.

* Step 6: Run the playbook, using the command below:

ansible-playbook nginx\_demo.yml

* Step 7: Check if Nginx is installed on the machine.

ps aux | grep nginx

2. What is the significance of the " notation? And how may variables or dynamic variable names be interpolated?

i think there is typo it should be {{ }} let us see below.

* One basic rule is to ‘always use {{}} except when:’. Conditionals are always run through Jinja2 as to resolve the expression. Therefore, ‘when:failed\_when:’ and ‘changed\_when:’ are always templated and we should avoid adding {{}}.
* In other cases, except when clause, we have to use brackets, otherwise, differentiating between an undefined variable and a string will be difficult to do.

3. What is the difference between an Ansible role and a playbook role?

* Playbooks:
* Playbooks are written in YAML format. YAML stands for Yet Another Markup Language.
* Playbooks are one of the core features of Ansible and tell Ansible what to execute.
* They are like a to do list for Ansible that contains a list of tasks.
* Ansible playbook is a script file which contains all the tasks that need to be performed along with all the ingredients required to perform these tasks. Roles are ways of automatically certain var files, tasks, and handlers based on the known file structure
* Roles:
* Roles contains certain vars\_files, tasks, and handlers based on which it had declared.
* In general grouping of tasks or handlers will allow the sharing of roles.

4. How can I write a multi-task Ansible handler in Ansible?

* Assume you want to create a handler that restarts a service only if it is already running.
* Tasks can notify handlers of generic topics, and handlers can listen to generic topics .
* This functionality makes it much easier to trigger multiple handlers.
* It also decouples handlers from their names, making it easier to share handlers among playbooks and roles

|  |
| --- |
| - name: Check if restarted  shell: check\_is\_started.sh  register: result  listen: Restart processes      - name: Restart conditionally step 2  service: name=service state=restarted  when: result  listen: Restart processes |

5. What are Ansible Vaults and how do you use them?

* Ansible Vault is used to encrypt and decrypt any structured data file used by Ansible. To use Ansible Vault, a command-line tool named ansible-vault is used to create, edit, encrypt, decrypt, and view files.
* Ansible Vault can encrypt any structured data file used by Ansible.
* This can include inventory variables, included variable files in a playbook, variable files passed as arguments when executing the playbook, or variables defined in Ansible roles.

Below is the poc:

* [ronak@ineuron\_poc ~]$ ansible-vault create secret.yml
* New Vault password: redhat
* Confirm New Vault password: redhat
* [ronak@ineuron\_poc ~]$ ansible-vault view secret1.yml

Vault password: **secret**

my\_secret: "yJJvPqhsiusmmPPZdnjndkdnYNDjdj782meUZcw"

* [ronak@ineuron\_poc ~]$ ansible-vault decrypt secret1.yml --output=secret1-decrypted.yml
* Vault password: redhat
* Decryption successful

6. How can I use Ansible to create encrypted files?

* The ansible-vault create command is used to create the encrypted file.
* Basic command to create a vault is

ansible-vault create vault.yml

* After typing this command it will ask for a password and then ask where to put your content. To check that the file has been encrypted, use the cat command.
* The following command is used to create encrypted files with --vault id.

ansible-vault create --vault-id password@prompt vault.yml

* Now suppose there is already a encrypted file and you want to edit it then use below command - ansible-vault edit secure.yml
* Similary we can decrpty a file with command ansible-vault decrypt secure.yml

7. What is Ansible Tower, exactly?

* Ansible Tower is basically an enterprise-level commercial product by RedHat. It is GUI base.
* It provides a web-based console and REST API to manage Ansible across teams in an organization.
* provides the user-interface to centrally execute and monitor Ansible playbooks across a complex enterprise IT environment
* Ansible Tower provides RBAC(role based access control) credentials management, delegation of jobs, compliance, and many other higher-level management features critical to a successful deployment of Ansible in an enterprise
* There are many features such as
* We can set up different dependencies among playbooks, or running multiple playbooks maintained by different teams at once
* if we want to know the status of any play or tasks can be monitored easily and also we can check what’s going to run next
* It helps to tracking logs are very important so that we can easily revert back to a previous state if something goes wrong.
* We can use the tower to run any command to a host or group of hosts in our inventory.
* There are so many other things that can be done with tower such as Job Scheduling, Notification Integration, Multi-Playbook Workflows ,CLI, etc.
* We can connect multiple Ansible Tower nodes into an Ansible Tower cluster as the clusters add redundancy and capacity, which allow you to scale Ansible automation across the enterprise.
* Integrated Notifications – This feature lets you notify a person or team when a job succeeds or fails across the entire organization at once, or customize on a per-job basis.

8. What are the benefits of the Ansible Tower?

* there are many benefits like below
* We can set up different dependencies among playbooks, or running multiple playbooks maintained by different teams at once
* if we want to know the status of any play or tasks can be monitored easily and also we can check what’s going to run next
* It helps to tracking logs are very important so that we can easily revert back to a previous state if something goes wrong.
* We can use the tower to run any command to a host or group of hosts in our inventory.
* There are so many other things that can be done with tower such as Job Scheduling, Notification Integration, Multi-Playbook Workflows ,CLI, etc.
* We can connect multiple Ansible Tower nodes into an Ansible Tower cluster as the clusters add redundancy and capacity, which allow you to scale Ansible automation across the enterprise.
* Integrated Notifications – This feature lets you notify a person or team when a job succeeds or fails across the entire organization at once, or customize on a per-job basis.

9. What is the role of Ansible in the Continuous Delivery pipeline? Explain.

* DevOps integrates development and operations.
* For today's modern test-driven applications, this integration is very important.
* With Ansible, both development and operations are provided with a stable environment, resulting in a smooth delivery pipeline.
* When developers begin to consider infrastructure as part of their applications, also known as Infrastructure as Code (IaC), stability and performance become the norm.
* Infrastructure as Code is the process of managing and provisioning computing infrastructure and their configuration through machine-processable definition files, rather than physical hardware configuration or the use of interactive configuration tools. A major advantage of Ansible Automation over its competitors is the ability to automate efficiently and effectively.
* In a Continuous Delivery pipeline, Sysadmins work tightly with developers, development velocity is improved, and more time is spent doing activities like performance tuning, experimenting, and getting things done, and less time is spent fixing problems.

10. Using Ansible, how do you build a LAMP stack and deploy a webpage?

* let us assume that want to deploy a website on 10 systems, so basic requirement would be every website deployment will require a OS, web-server to host web application, Database to store data for app and PHP.
* So We use ansible playbook to install these prerequisites on all 10 systems at once.
* For this you can use two virtual machines, one as a server where Ansible is installed and the other machine acts as the remote host.
* create a simple static webpage and let's save in a folder index which has two files index.html and style.css
* In the below code creates a single Ansible playbook to install Apache, MySql, and PHP:
* **LAMP.yml**

|  |
| --- |
| ---  # demo to setup LAMP Stack  - hosts: host1  tasks:    - name: Add ppa repository  become: yes  apt\_repository: repo=ppa:ondrej/php    - name: Install lamp stack  become: yes  apt:  pkg:  - apache2  - mysql-server  - php7.0  - php7.0-mysql  state: present  update cache: yes    - name: start apache server  become: yes  service:  name: apache2  state: started  enabled: yes    - name: start mysql service  become: yes  services:  name: mysql  state: started  enabled: yes    - name: create target directory  file: path=/var/www/html state=directory mode=0755    - name: deploy index.html  became: yes  copy:  src: /etc/ansible/index/index.html  dest: var/www/html/index/index.html |

* Now, there are different 6 main tasks with some function:
* first task will adds the repository required to install MySQL and PHP.
* second task will installs apache2, MySQL-server, PHP, and PHP-MySQL.
* third and fourth task will starts the Apache and MySQL service.
* fifth task will creates a target directory in the host machine
* sixth task will executes the index.html file, it picks up the file from the server machine and copies it into the host machine.
* Run below command to run the playbook

ansible-playbook lamp.yml -K

* Because the playbook requires sudo to run, we’re also including the -K argument to provide the remote user’s sudo password when prompted by Ansible: