Ansible Interview Questions

Ansible is a software management tool that works on Python. It is used to set up, manage and deploy an application which uses SSH without any downtime. Ansible is also a configuration management system used for managing IT infrastructure and deploying software apps to remote nodes. Ansible only connects through SSH, remote PowerShell or some other APIs. In simpler words. Ansible can help you with configuration management, task automation, and application deployment.

There are two types of servers in **Ansible**: controlling machines and the nodes. In controlling machines, orchestrate begins. These machines tell the location of the nodes. Nodes are managed by controlling machines over SSH. One of the reasons why Ansible is more preferred over other configuration management software like Puppet, Chef is because Ansible uses an agentless architecture. Under this system, nodes are not required to install and run daemons to connect with a controlling machine whereas agent architecture requires that nodes must have locally installed daemons to connect with controlling machines.

Over the years, there has been a great demand for Ansible experts and analytics and if you're planning to sit for an interview for Ansible, then you're at right place. We have listed few important questions that you should be well aware of before going ahead with the interview.

Read Best Ansible interview questions and answers from below:

Q1. What is Ansible?

Ansible is an open source automation platform which can help you with configuration management, task automation and application deployment. Ansible uses SSH installed on all systems unlike other configuration softwares that work on agent architecture. Ansible also do IT orchestration where you run tasks and create a chain of events that happen on different servers and devices. It is written on Python language which needs to be installed on the remote host. Ansible is very easy to set up yet it is a very powerful tool for software deployment.

Q2. Talk about Ansible architecture.

Ansible works on 'agentless architecture'. It works by connecting to your nodes and pushes out Ansible modules to them which are small programmes. These modules are written to the resource nodes of the desired state of the system. With the help of SSH, Ansible then executes these modules and removes them when done. Since ansibles are based on agentless architecture, your pool of modules can dwell on any machine without requiring any server, daemons or databases. You just require terminal program, text editor and a version control system to keep a check on changes to your content. Ansible's "authorized_key" is used to give directions as to what machines will use which hosts.

Q3. List some advantages of using Ansible.

Unlike other configuration management system, Ansible is the most sought after software applications these days. it offers the following benefits to its users:

- Agentless- Its work structure makes use of agentless architecture. The nodes are not required to install and run background daemons to connect with a controlling machine.
- Low overhead- due to agentless model, Ansibles reduces the overheads on the network by preventing the nodes from polling the controlling machine.
- Secure and consistent- Ansible only uses SSH and Python on the managed nodes. This ensures safety and security. Also, Ansible ensures consistent environments.
- Reliable- an Ansible playbook can be idempotent when written carefully. This prevents unexpected side-effects on the managed systems.
- Good performance- Ansible delivers flawless performance. Though it is very easy to set up yet it is a powerful tool for deploying software applications using SSH.

Q4. Describe the working of Ansible.

Ansible consists of nodes and controlling machine. The controlling machine is the place where Ansibles are installed and nodes are managed by these machines over SSH. With help of SSH protocols, controlling machines deploy modules which are temporarily stored on remote nodes. They will then communicate with the Ansible machine over standard output through JSON connection. Since Ansible is agentless, it controls more than 100 nodes using SSH and the entire operation can be managed using one single command ansible. In cases where you have to build multiple commands, you have to create playbooks in YAML file format. Playbooks are nothing but a bunch of commands which can perform numerous tasks.

Q5. What is Ansible galaxy?

Ansible role is a way of bundling automation content and making it reusable. To share such Ansible roles, Ansible galaxy is required. In other words, Ansible galaxy is a tool bundled with Ansible which is used to create base directory structure. Using the Ansible command, Ansible can communicate with configured clients. This also facilitates automate configuration with the help of Ansible playbook command.

Q6. What is the use of Ansible?

Ansible can be used to deploy many software applications to many nodes using a single command. This requires a little technical know-how on behalf of the user. In IT infrastructures, Ansibles are used to manage and deploy software applications to remote nodes. Let’s say you want to deploy a single software or multiple softwares into more than 100 nodes using a single command, here Ansible comes to use.

Q7. Give a comparison between Ansible and puppet.

A comparative study between Ansible and puppet is given below:

Ansible: Ansible is very simple to set up. It is a simple technology which is written in YAML language. It is based on agent-less architecture which doesn't require nodes to locally install daemons. It facilitates automated workflow for continuous and hassle free delivery. Ansible doesn't support windows. It comes with good GUI and CLI accepts command in almost every language.

Puppet: puppet is a complex technology as compared with Ansible. It is written in Ruby language. It works on easy installation and facilitates visualisation and reporting. It is not based on agentless architecture and unlike Ansible, puppet supports for almost all major operating systems. The prerequisite for using puppet is that the user must learn the puppet DSL language.

Q8. What is Ansible tower?

Ansible is a web-based hub for all your automation tasks. It is based on agentless model that doesn't require nodes to have locally installed daemons to connect with controlling machine. The Ansible tower is free for usage till 10 nodes.

Q9. Enlist the differences between variable name and environment variables.

Variable Name	Environment variables
To build variable names, you need to add strings.	To access environment variables, you have to access existing variables.
It allows adding more strings.	We need to see advances playbook section to set environment variables.
We use the ipv4 address for variable names.	We use {{ ansible_env.SOME_VARIABLE }} for remote environment variables.
{{ hostvars[inventory_hostname][‘ansible_’ + which_interface][‘ipv4?][‘address’]	# … vars: local_home: ”
}}	{{ lookup(‘env’,’HOME’) }}”

Q10. Explain different modules in Ansible.

There are two types of modules in Ansible namely core modules and extra modules. Modules in Ansible are idempotent and the clients can perform the same result by using modules in Ansible for the operation to be idempotent.

Core modules– The Ansible team gives more importance to these modules over extra modules. Core modules are always shipped with Ansible software.

Extra modules– These modules are maintained by Ansible community and are reusable but receive a lower rate of response to issues. These modules are bundled with Ansible but can be separately available in future.

Q11. When to use {{}} ? How to interpolate variables or dynamic variable names?

- A steadfast rule is 'always use {{ }} except when *when:*'. Conditionals are always run through Jinja2 as to resolve the expression, so when: failed_when: and changed_when: are always templates and you should avoid adding {{}}.
- It is always recommended to use brackets even if you have previously used variables without specifying (like *with*_clauses). This makes it hard to distinguish between a string and an undefined variable.
- Another rule is 'moustaches don't stack'. We often see this:
- {{ somevar_{{other_var}} }}
- The above DOES NOT WORK, if you need to use a dynamic variable use the hostvars or vars dictionary as appropriate:
- {{ hostvars[inventory_hostname]['somevar_' + other_var] }}

Q12. How to install Ansible?

The most effective way to install Ansible for Ubuntu is to add project's personal package archive (PPA) to your system. For this, you have to install the software properties common package. This will ensure that you can work with PPA easily. On older versions, this package was called as python software properties. Once the package has been installed, type the following command to add the Ansible PPA.

```
sudo apt-add-repository ppa:ansible/ansible
```

Press enter for PPA addition. Once done, refresh the package to see available PPA packages and you can install the software.

```
sudo apt-get install ansible
sudo apt-get update
```

We have the software required to administer our servers through Ansible.

Q13. How to generate crypto passwords for the user module?

To generate crypto passwords for the user module, mkpasswd utility available in Linux systems is a great option.

```
Mkpasswd --method=sha-512.
```

However, if you're OS X user and this utility is not installed on your system, you can still generate it using python. First check that Passlib passwo4rd hashing library is installed. Once the library is installed, SHA512 password values can then be generated as follows:

```
python -c "from passlib.hash import sha512_crypt; import getpass; print sha512_c
```

To generate the hashed version of the password, use the hashing filters. Also, use vault to protect sensitive data.

Q14. What is Ansible role?

The very first step in creating an Ansible role is creating its directory structure. To create the base directory structure, use a tool bundled with Ansible called Ansible galaxy.

```
$ ansible-galaxy init azavea.packer
azavea.packer was created successfully
```

This command will create an azavae.packer directory.

Q15. What is the way to access shell environment variables in Ansible?

The user needs to use the ‘env’ lookup plugin in order to access the existing variables. If you want to access the value of the office environment on the management machine, you need to write the following code.

1. _ _ _
2. # . . .
3. vars :
4. local_home : “{{ lookup(‘env’,’Office’) }}
5. I
6. {{ ansible_env . SOME_VARIABLE }}

Q16. How to access a variable of the first host in a group?

The trick lies in this command:

```
{{ hostvars[groups['webservers'][0]]['ansible_eth0']['ipv4']['address'] }}
```

Here, we’re pulling out the hostname of the first machine of the webservers group. If you’re doing this in a template, use the Jinja2 ‘#set’ or in a playbook, you can also use set_fact:

```
- set_fact: headnode={{ groups[['webservers'][0]] }}
- debug: msg={{ hostvars[headnode].ansible_eth0.ipv4.address }}
```

Q17. When should you test playbooks and roles?

In Ansible, tests are added either in new playbooks or to the existing ones. So, most of the testing job offers clear hosting each time and with this method, you need to make very minor changes to coding.

Q18. How to keep secret data in playbook?

If you want to keep secret data in your ansible content and still share it publicly, then you can use Vault in playbooks. If you're using –v (verbose) mode, and don't want to show the results, then following command can be used:

```
name: secret task
shell: /usr/bin/do_something --value={{ secret_value }}
no_log: True
```

This can be used to hide sensitive data from others.

Q19. What is the method to check the inventory vars defined for the host?

use the following command to check the inventory:

```
ansible -m debug -a "var=hostvars['hostname']" localhost
```

Q20. What are ad-hoc commands?

Ad-hoc commands are a way for us to take actions on our hosts without having us to write the playbooks. If you want to reboot all the hosts in a particular group, then you can either write a new playbook or simply run a one-off ad-hoc command.