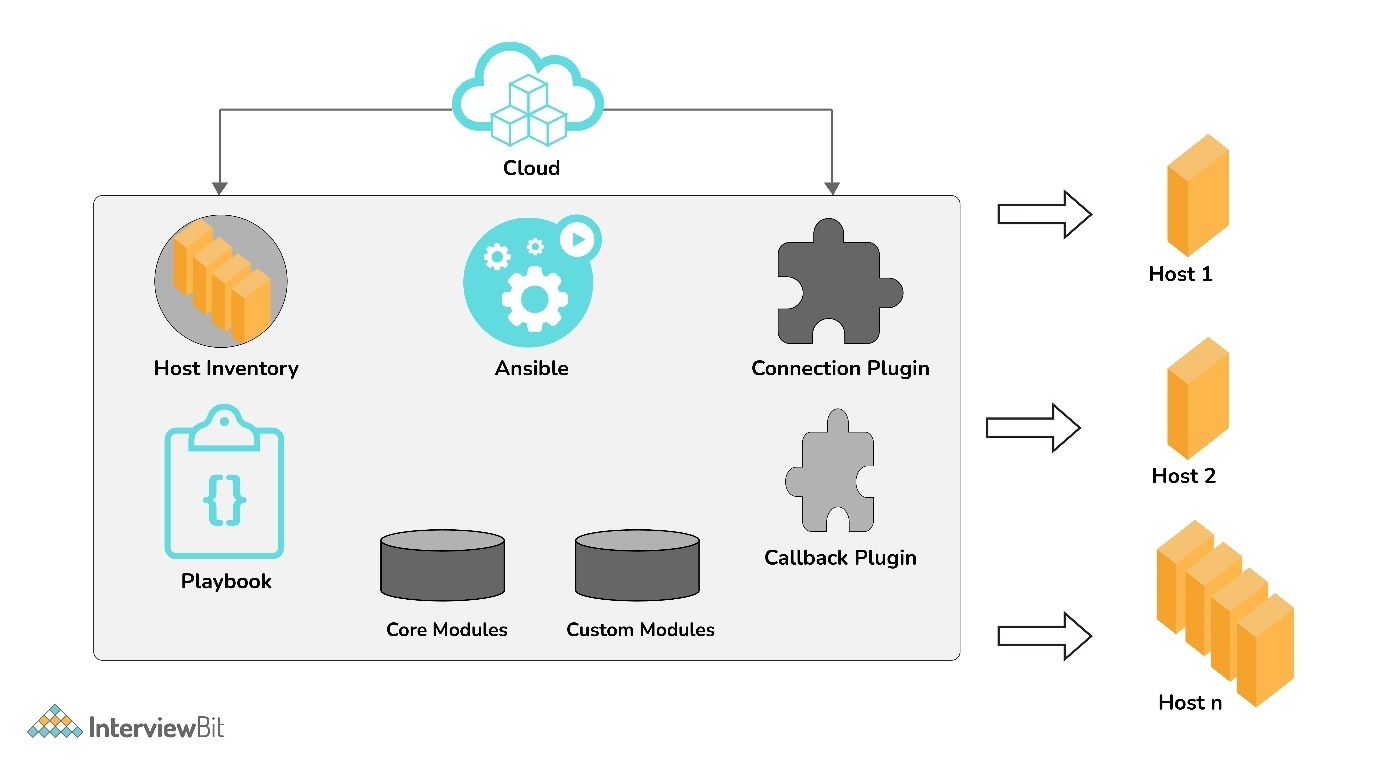
**1. What is Configuration Management?**

It’s a practice that we should follow in order to keep track of all updates that are going into the system over a period of time. This also helps in a situation where a major bug has been introduced to the system due to some new changes and we need to fix it with minimum downtime. Instead of fixing the bug, we can roll back the new changes(which caused this bug) as we have been tracking those.

**3. How does Ansible work?**

Ansible is a combination of multiple pieces working together to become an automation tool. Mainly these are modules, playbooks, and plugins.

* Modules are small codes that will get executed. There are multiple inbuilt modules that serve as a starting point for building tasks.
* Playbooks contain plays which further is a group of tasks. This is the place to define the workflow or the steps needed to complete a process
* Plugins are special kinds of modules that run on the main control machine for logging purposes. There are other types of plugins also.



The playbooks ran via an Ansible automation engine. These playbooks contain modules that are basically actions that run in host machines. The mechanism is followed here is the push mechanism, so ansible pushes small programs to these host machines which are written to be resource models of the desired state of the system.

### 4. What are the features of Ansible?

It has the following features:

* **Agentless** – Unlike puppet or chef there is no software or agent managing the nodes.
* **Python** – Built on top of python which is very easy to learn and write scripts and one of the robust programming languages.
* **SSH**– Passwordless network authentication which makes it more secure and easy to set up.
* **Push architecture** – The core concept is to push multiple small codes to the configure and run the action on client nodes.
* **Setup** – This is very easy to set up with a very low learning curve and any open source so that anyone can get hands-on.
* **Manage Inventory** – Machines’ addresses are stored in a simple text format and we can add different sources of truth to pull the list using plugins such as Openstack, Rackspace, etc.

### 5. Explain Infrastructure as Code?

Infrastructure as Code or IaC is a process that DevOps teams should follow to have a more organized way of managing the infra. Instead of some throwaway scripts or manually configuring any cloud component, there should be a code repo where all of these will lie and any change in configuration should be done through it. It is wise to put it under source control also. This improves speed, consistency, and accountability.

### 6. What is Ansible Galaxy?

Galaxy is a repository of Ansible roles that can be shared among users and can be directly dropped into playbooks for execution. It is also used for the distribution of packages containing roles, plugins, and modules also known as collection. The ansible-galaxy-collection command implements similar to init, build, install, etc like an ansible-galaxy command.

### 7. Explain Ansible modules in detail?

Ansible modules are like functions or standalone scripts which run specific tasks idempotently. The return value of these are JSON string in stdout and input depends on the type of module. These are used by Ansible playbooks.  
There are 2 types of modules in Ansible:

* **Core Modules**

The core Ansible team is responsible for maintaining these modules thus these come with Ansible itself. The issues reported are fixed on priority than those in the “extras” repo.

* **Extras Modules**

The Ansible community maintains these modules so, for now, these are being shipped with Ansible but they might get discontinued in the future. These can be used but if there are any feature requests or issues they will be updated on low priority.

Now popular extra modules might enter into the core modules anytime. You may find these separate repos for these modules as ansible-modules-core and ansible-modules-extra respectively.

### 8. What is a YAML file and how do we use it in Ansible?

YAML or files are like any formatted text file with few sets of rules just like JSON or XML. Ansible uses this syntax for playbooks as it is more readable than other formats.  
An example of JSON vs YAML is:

{

"object": {

"key": "value",

"array": [

{

"null\_value": null

},

{

"boolean": true

},

{

"integer": 1

},

{

"alias": "aliases are like variables"

}

]

}

}

---

object:

key: value

array:

- null\_value:

- boolean: true

- integer: 1

- alias: aliases are like variables

### 9. What are Ansible tasks?

The task is a unit action of Ansible. It helps by breaking a configuration policy into smaller files or blocks of code. These blocks can be used in automating a process. For example, to install a package or update a software

Install <package\_name>, update <software\_name>

### 10. How to use YAML files in high programming languages such as JAVA, Python, etc?

YAML is supported in most programming languages and can be easily integrated with user programs.  
In JAVA we can use the Jackson module which also parses XML and JSON. For e.g

// We need to declare Topic class with necessary attributes such as name, total\_score, user\_score, sub\_topics

List<Topic> topics = **new** ArrayList<Topic>();

topics.add(**new** Topic("String Manipulation", 10, 6));

topics.add(**new** Topic("Knapsack", 5, 5));

topics.add(**new** Topic("Sorting", 20, 13));

// We want to save this Topic in a YAML file

Topic topic = **new** Topic("DS & Algo", 35, 24, topics);

// ObjectMapper is instantiated just like before

ObjectMapper om = **new** ObjectMapper(**new** YAMLFactory());

// We write the `topic` into `topic.yaml`

om.writeValue(**new** File("/src/main/resources/topics.yaml"), topic);

---

name: "DS & Algo"

total\_score: 35

user\_score: 24

sub\_topics:

- name: "String Manipulation"

total\_score: 10

user\_score: 6

- name: "Knapsack"

total\_score: 5

user\_score: 5

- name: "Sorting"

total\_score: 20

user\_score: 13

Similarly, we can read from YAML also:

// Loading the YAML file from the /resources folder

ClassLoader classLoader = Thread.currentThread().getContextClassLoader();

File file = **new** File(classLoader.getResource("topic.yaml").getFile());

// Instantiating a new ObjectMapper as a YAMLFactory

ObjectMapper om = **new** ObjectMapper(**new** YAMLFactory());

// Mapping the employee from the YAML file to the Employee class

Topic topic = om.readValue(file, Topic.class);

In python similarly, we can use the pyyaml library and read and write easily in YAML format.

## Intermediate Ansible Interview Questions

### 11. How to setup a jump host to access servers having no direct access?

First, we need to set a ProxyCommand in ansible\_ssh\_common\_args inventory variable, since any arguments specified in this variable are added to the sftp/scp/ssh command line when connecting to the relevant host(s). For example

[gatewayed]

staging1 ansible\_host=10.0.2.1

staging2 ansible\_host=10.0.2.2

To create a jump host for these we need to add a command in ansible\_ssh\_common\_args

ansible\_ssh\_common\_args: '-o ProxyCommand="ssh -W %h:%p -q user@gateway.example.com"'

In this way whenever we will try to connect to any host in the gatewayed group ansible will append these arguments to the command line.

### 12. How to automate the password input in playbook using encrypted files?

To automate password input we can have a password file for all the passwords of encrypted files will be saved and ansible can make a call to fetch those when required.

ansible\_ssh\_common\_args: '-o ProxyCommand="ssh -W %h:%p -q user@gateway.example.com"'

This can also be achieved by having a separate script that specifies the passwords. But in this case, we need to print a password to stdout to work without annoying errors.

ansible-playbook launch.yml --vault-password-file ~/ .vault\_pass.py

### 13. What are callback plugins in Ansible?

Callback plugins basically control most of the output we see while running cmd programs. But it can also be used to add additional output. For example log\_plays callback is used to record playbook events to a log file, and mail callback is used to send email on playbook failures. We can also add custom callback plugins by dropping them into a callback\_plugins directory adjacent to play, inside a role, or by putting it in one of the callback directory sources configured in ansible.cfg.

### 14. What is Ansible Inventory and its types?

In Ansible, there are two types of inventory files: Static and Dynamic.

* **Static inventory** file is a list of managed hosts declared under a host group using either hostnames or IP addresses in a plain text file. The managed host entries are listed below the group name in each line. For example

[gatewayed]

staging1 ansible\_host=10.0.2.1

staging2 ansible\_host=10.0.2.2

* **Dynamic inventory** is generated by a script written in Python or any other programming language or by using plugins(preferable). In a cloud setup, static inventory file configuration will fail since IP addresses change once a virtual server is stopped and started again. We create a demo\_aws\_ec2.yaml file for the config such as

plugin: aws\_ec2 regions:

ap-south-1 filters:

tag:tagtype: testing

Now we can fetch using this command

ansible-inventory -i demo\_aws\_ec2.yaml -graph

### 15. What is Ansible Vault?

Ansible vault is used to keep sensitive data such as passwords instead of placing it as plaintext in playbooks or roles. Any structured data file or any single value inside the YAML file can be encrypted by Ansible.

To encrypt a file

ansible-vault encrypt foo.yml bar.yml baz.yml

And similarly to decrypt

ansible-vault decrypt foo.yml bar.yml baz.yml

### 16. How can looping be done over a list of hosts in a group, inside of a template?

This can be done by accessing the “$groups” dictionary in the template, like so:

{% for host in groups['db\_servers'] %}

{{ host }}

{% endfor %}

If we need to access facts also we need to make sure that the facts have been populated. For instance, a play that talks to db\_servers:

- hosts: db\_servers

tasks:

- debug: msg="Something to debug"

Now, this can be used within a template, like so:

{% for host in groups['db\_servers'] %}

{{ hostvars[host]['ansible\_eth0']['ipv4']['address'] }}

{% endfor %}.

### 17. What is the ad-hoc command in Ansible?

Ad-hoc commands are like one-line playbooks to perform a specific task only. The syntax for the ad-hoc command is

ansible [pattern] -m [module] -a "[module options]"

For example, we need to reboot all servers in the staging group

ansible atlanta -a "/sbin/reboot" -u username --become [--ask-become-pass]

### 18. Install Nginx using Ansible playbook?

The playbook file would be:

- hosts: stagingwebservers

gather\_facts: False

vars:

- server\_port: 8080

tasks:

- name: install nginx

apt: pkg=nginx state=installed update\_cache=true

- 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

- name: restart flask app

service: name=flask-demo state=restarted

...

In the above playbook, we are fetching all hosts of stagingwebservers group for executing these tasks. The first task is to install Nginx and then configure it. We are also taking a flask server for reference. In the end, we also defined handlers so that in case the state changes it will restart Nginx. After executing the above playbook we can verify whether Nginx is installed or not.

ps waux | grep nginx

### 19. How do I access a variable name programmatically?

Variable names can be built by adding strings together. For example, if we need to get ipv4 address of an arbitrary interface, where the interface to be used may be supplied via a role parameter or other input, we can do it in this way.

{{ hostvars[inventory\_hostname]['ansible\_' + which\_interface]['ipv4']['address'] }}

### 20. What is the difference between Ansible and Puppet?

**Management and Scheduling:**  In Ansible, the server pushes the configuration to the nodes on the other hand in puppet, the client pulls the configuration from the server. Also for scheduling, the puppet has an agent who polls every 30mins(default settings) to make sure all nodes are in a desirable state. Ansible doesn’t have that feature in the free version.  
**Availability:** Ansible has backup secondary nodes and puppet has more than one master node. So both try to be highly available.  
**Setup:** Puppet is considered to be harder to set up than ansible as it has a client-server architecture and also there’s a specific language called Puppet DSL which is its own declarative language.

### 21. What is Ansible Tower and what are its features?

Ansible Tower is an enterprise-level solution by RedHat. It provides a web-based console and REST API to manage Ansible across teams in an organization. There are many features such as

* Workflow Editor - We can set up different dependencies among playbooks, or running multiple playbooks maintained by different teams at once
* Real-Time Analysis - The status of any play or tasks can be monitored easily and we can check what’s going to run next
* Audit Trail - Tracking logs are very important so that we can quickly revert back to a functional state if something bad happens.
* Execute Commands Remotely - We can use the tower to run any command to a host or group of hosts in our inventory.

There are other features also such as Job Scheduling, Notification Integration, CLI, etc.

### 22. Explain how you will copy files recursively onto a target host?

There’s a copy module that has a recursive parameter in it but there’s something called synchronize which is more efficient for large numbers of files.

For example:

- synchronize:

src: /first/absolute/path

dest: /second/absolute/path

delegate\_to: "{{ inventory\_hostname }}"

### 23. What is the best way to make Content Reusable/ Redistributable?

To make content reusable and redistributable Ansible roles can be used. Ansible roles are basically a level of abstraction to organize playbooks. For example, if we need to execute 10 tasks on 5 systems, writing all of them in the playbook might lead to blunders and confusion. Instead we create 10 roles and call them inside the playbook.

### 24. What are handlers?

Handlers are like special tasks which only run if the Task contains a “notify” directive.

tasks:

- name: install nginx

apt: pkg=nginx state=installed update\_cache=true

notify:

- start nginx

handlers:

- name: start nginx

service: name=nginx state=started

In the above example after installing NGINX we are starting the server using a `start nginx` handler.

### 25. How to generate encrypted passwords for a user module?

Ansible has a very simple ad-hoc command for this

ansible all -i localhost, -m debug -a "msg={{ 'mypassword' | password\_hash('sha512', 'mysecretsalt') }}"

We can also use the Passlib library of Python, e.g

python -c "from passlib.hash import sha512\_crypt; import getpass; print(sha512\_crypt.using(rounds=5000).hash(getpass.getpass()))"

On top of this, we should also avoid storing raw passwords in playbook or host\_vars, instead, we should use integrated methods to generate a hash version of a password.

### 26. How does dot notation and array notation of variables are different?

Dot notation works fine unless we stump upon few special cases such as

* If the variable contains a dot(.), colon(:), starting or ending with an underscore or any known public attribute.
* If there’s a collision between methods and attributes of python dictionaries.
* Array notation also allows for dynamic variable composition.

## Advanced Ansible Interview Questions

### 27. How does Ansible synchronize module works?

Ansible synchronize is a module similar to rsync in Linux machines which we can use in playbooks. The features are similar to rsync such as archive, compress, delete, etc but there are few limitations also such as

* Rsync must be installed on both source and target systems
* Need to specify delegate\_to to change the source from localhost to some other port
* Need to handle user permission as files are accessible as per remote user.
* We should always give the full path of the destination host location in case we use sudo otherwise files will be copied to the remote user home directory.
* Linux rsync limitations related to hard links are also applied here.
* It forces -delay-updates to avoid the broken state in case of connection failure

An example of synchronize module is

---

- hosts: host-remote tasks:

- name: sync from sync\_folder

synchronize:

src: /var/tmp/sync\_folder dest: /var/tmp/

Here we are transferring files of /var/tmp/sync\_folder folder to remote machine’s /var/tmp folder

### 28. How does the Ansible firewalld module work?

Ansible firewalld is used to manage firewall rules on host machines. This works just as Linux firewalld daemon for allowing/blocking services from the port. It is split into two major concepts

* **Zones:**This is the location for which we can control which services are exposed to or a location to which one the local network interface is connected.
* **Services:** These are typically a series of port/protocol combinations (sockets) that your host may be listening on, which can then be placed in one or more zones

Few examples of setting up firewalld are

- name: permit traffic in default zone for https service

ansible.posix.firewalld:

service: https

permanent: yes

state: enabled

- name: do not permit traffic in default zone on port 8081/tcp

ansible.posix.firewalld:

port: 8081/tcp

permanent: yes

state: disabled

### 29. How is the Ansible set\_fact module different from vars, vars\_file, or include\_var?

 In Ansible, set\_fact is used to set new variable values on a host-by-host basis which is just like ansible facts, discovered by the setup module. These variables are available to subsequent plays in a playbook. In the case of vars, vars\_file, or include\_var we know the value beforehand whereas when using set\_fact, we can store the value after preparing it on the fly using certain tasks like using filters or taking subparts of another variable. We can also set a fact cache over it.

set\_fact variable assignment is done by using key-pair values where the key is the variable name and the value is the assignment to it. A simple example will be like below

- set\_fact:

one\_fact: value1

second\_fact:

value2

### 30. When is it unsafe to bulk-set task arguments from a variable?

All of the task's arguments can be dictionary-typed variables which can be useful in some dynamic execution scenarios also. However, Ansible issues a warning since it introduces a security risk.

vars:

usermod\_args:

name: testuser

state: present

update\_password: always

tasks:

- user: '{{ usermod\_args }}'

In the above example, the values passed to the variable usermod\_args could be overwritten by some other malicious values in the host facts on a compromised target machine. To avoid this

* bulk variable precedence should be greater than host facts.
* need to disable INJECT\_FACTS\_AS\_VARS configuration to avoid collision of fact values with variables.

### 31. Explain Ansible register.

Ansible register is used to store the output from task execution in a variable. This is useful when we have different outputs from each remote host. The register value is valid throughout the playbook execution so we can make use of set\_fact to manipulate the data and provide input to other tasks accordingly.

- hosts: all tasks:

name: find all txt files in /home shell: "find /home -name \*.txt" register: find\_txt\_files

debug:

var: find\_txt\_files

In the above example, we are searching for all .txt files in the remote host’s home folder and then capturing it in find\_txt\_files and displaying that variable.

### 32. How can we delegate tasks in Ansible?

Task delegation is an important feature of Ansible since there might be use cases where we would want to perform a task on one host with reference to other hosts. We can do this using the delegate\_to keyword.

For example, if we want to manage nodes in a load balancer pool we can do:

- hosts: webservers

serial: 5

tasks:

- name: Take machine out of ELB pool

ansible.builtin.command: /usr/bin/take\_out\_of\_pool {{ inventory\_hostname }}

delegate\_to: 127.0.0.1

- name: Actual steps would go here

ansible.builtin.yum:

name: acme-web-stack

state: latest

- name: Add machine back to ELB pool

ansible.builtin.command: /usr/bin/add\_back\_to\_pool {{ inventory\_hostname }}

delegate\_to: 127.0.0.1

We are also defining serial to control the number of hosts executing at one time. There is another shorthand syntax called local\_action which can be used instead of delegate\_to.

...

tasks:

- name: Take machine out of ELB pool

local\_action: ansible.builtin.command /usr/bin/take\_out\_of\_pool {{ inventory\_hostname }}

...

But there are few exceptions also such as include, add\_host, and debug tasks that cannot be delegated.

### ****1. What is Ansible?****

Ansible is a configuration management system. It is used to set up and manage infrastructure and applications. It allows users to deploy and update applications using SSH, without needing to install an agent on a remote system.

### ****2. What is the use of Ansible?****

Ansible is used for managing IT infrastructure and deploying software apps to remote nodes. Ansible allows you to deploy an application to many nodes with one single command. However, for that, there is a need for some programming knowledge to understand the Ansible scripts.

### ****3. What are the features of Ansible?****

Ansible has the following features:

* **Agentless:** Unlike Puppet or Chef, there is no software or agent managing the nodes
* **Python:** Built on top of Python, which is very easy to learn and write scripts. It is one of the robust programming languages
* **SSH:** Passwordless network authentication makes it more secure and easy to set up
* **Push architecture:** The core concept is to push multiple small codes to configure and run the action on client nodes
* **Set up:** This is very easy to set up with a very low learning curve. It is open-source; so, anyone can access it.
* **Manage inventory:** Machines’ addresses are stored in a simple text format and we can add different sources of truth to pull the list using plug-ins such as OpenStack, Rackspace, etc.

### ****4. What are the advantages of Ansible?****

Ansible has many strengths which include:

* It is agentless and only requires SSH service running on target machines.
* Python is the only required dependency and, fortunately, most systems come with it pre-installed.
* It requires minimal resources; so, there is low overhead.
* It is easy to learn and understand since Ansible tasks are written in YAML.
* Unlike other tools, most of which are procedural, Ansible is declarative; it defines the desired state and fulfills the requirements needed to achieve it.

### ****5. What is Ansible Galaxy?****

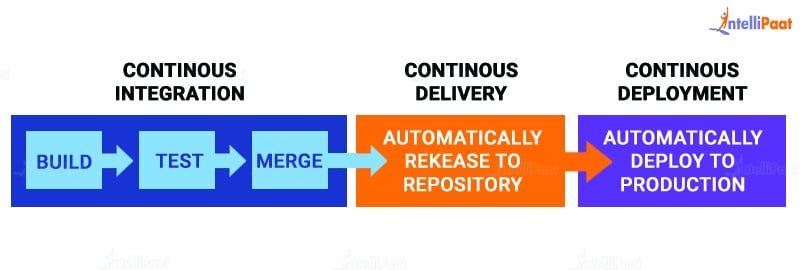
Ansible can communicate with configured clients from the command line by using the Ansible command. It also allows you to automate configuration by using the Ansible-playbook command. To create the base directory structure, you can use a tool bundled with Ansible, which is known as ansible-galaxy.

**Command:** *ansible-galaxy init azavea. Packer*

### ****6. What is CI/CD?****

Continuous integration is something that is used for streamlining the development and deployment process. This has led to the more rapid development of cohesive software. Each integration is verified by an automated build to detect integration errors as quickly as possible.

Continuous delivery is the process where your code after being pushed to a remote repository can be taken to production at any time. It is, in simpler words, a process where you build software in such a way that it can be released to production at any time.



### ****7. What is configuration management?****

It is a practice that we should follow in order to keep track of all updates that are going into the system over a period of time. This also helps in a situation where a major bug has been introduced to the system due to some new changes that need to be fixed with minimum downtime. Configuration management (CM) keeps a track of all updates that are needed in a system and it ensures that the current design and build state of the system is up to date and functioning correctly.

Bottom of Form

### ****8. What are Ansible server requirements?****

If you are a Windows user, then you need to have a virtual machine in which Linux should be installed. It requires Python 2.6 version or higher. If these requirements are fulfilled, then you can proceed with ease.

### ****9. What are Ansible tasks?****

The task is a unit action of Ansible. It helps by breaking a configuration policy into smaller files or blocks of code. These blocks can be used in automating a process. For example, to install a package or update a software:

**Command:***Install <package\_name>*

**Command:***update <software\_name>*

### ****10. Explain a few of the basic terminologies or concepts in Ansible****

A few of the basic terms that are commonly used while operating on Ansible are:

* **Controller machine:** The controller machine is responsible for provisioning servers that are being managed. It is the machine where Ansible is installed.
* **Inventory:** An inventory is an initialization file that has details about the different servers that you are managing.
* **Playbook:** It is a code file written in the YAML format. A playbook basically contains the tasks that need to be executed or automated.
* **Task:** Each task represents a single procedure that needs to be executed, e.g., installing a library.
* **Module:** A module is a set of tasks that can be executed. Ansible has hundreds of built-in modules but you can also create custom ones.
* **Role:** An Ansible role is a predefined way for organizing playbooks and other files in order to facilitate sharing and reusing portions of provisioning.
* **Play:** A task executed from start to finish or the execution of a playbook is called a play.
* **Facts:** Facts are global variables that store details about the system such as network interfaces or operating systems.
* **Handlers:** Handlers are used to trigger the status of a service such as restarting or stopping a service.

### ****11. What is a playbook?****

A playbook has a series of YAML-based files that send commands to remote computers via scripts. Developers can configure complete complex environments by passing a script to the required systems rather than using individual commands to configure computers from the command line remotely. Playbooks are one of Ansible’s strongest selling points and are often referred to as Ansible’s building blocks.

### ****12. State the differences between variable names and environment variables****

|  |  |
| --- | --- |
| **Variable Names** | **Environment Variables** |
| It can be built by adding strings. | To access the environment variable, the existing variables need to be accessed. |
| {{ hostvars[inventory\_hostname][‘ansible\_’ + which\_interface][‘ipv4’][‘address’] }} | # … vars: local\_home: “{{ lookup(‘env’,’HOME’) }}” |
| You can easily create multiple variable names by adding strings. | To set environment variables, you need to see the advanced playbooks section. |
| Ipv4 address type is used for variable names. | For remote environment variables, use {{ ansible\_env.SOME\_VARIABLE }}. |

### ****13. Where are tags used?****

A tag is an attribute that sets the Ansible structure, plays, tasks, and roles. When an extensive playbook is needed, it is more useful to run just a part of it as opposed to the entire thing. That is where tags are used.

### ****14. Which protocol does Ansible use to communicate with Linux and Windows?****

For Linux, the protocol used is SSH.

For Windows, the protocol used is WinRM.

### ****15. What are ad hoc commands? Give an example****

Ad hoc commands are simple one-line commands used to perform a certain task. You can think of ad hoc commands as an alternative to writing playbooks. An example of an ad hoc command is as follows:

**Command:***ansible host -m netscaler -a "nsc\_host=nsc.example.com user=apiuser password=apipass"*

### ****16. Compare Ansible with Chef****

|  |  |
| --- | --- |
| **Ansible** | **Chef** |
| Easy to set up | Not very easy to set up |
| Easy to manage | Management is not easy |
| Configuration language is YAML (Python) | Configuration language is DSL (Ruby) |
| Self-support package is $5,000 annually.  Premium version costs $14,000 annually for each 100 nodes | Standard plan starts at $72 annually per node. The automation version charges $137 per node annually |

### ****17. What is a YAML file and how do we use it in Ansible?****

YAML files are like any formatted text file with a few sets of rules similar to that of JSON or XML. Ansible uses this syntax for playbooks as it is more readable than other formats.

### ****18. Code difference between JSON and YAML:****

**JSON:**

*{*

*"object": {*

*"key": "value",*

*"array": [*

*{*

*"null\_value": null*

*},*

*{*

*"boolean": true*

*},*

*{*

*"integer": 1*

*},*

*{*

*"alias": "aliases are like variables"*

*}*

*]*

*}*

*}*

**YAML:**

*---*

*object:*

*key: value*

*array:*

*- null\_value:*

*- boolean: true*

*- integer: 1*

*- alias: aliases are like variables*

### ****19. How is Ansible different from Puppet?****

|  |  |
| --- | --- |
| **Ansible** | **Puppet** |
| Easy to set up | Comparatively harder to set up |
| Very easy to manage | Not very easy to manage |
| Configuration language is YAML (Python) | Configuration language is DSL (Puppet DSL) |
| Self-support package is $5,000 annually. The premium version costs $14,000 annually for each 100 nodes | Enterprise pricing starts at $120 for every node annually. The premium version costs $19,900 annually for each 100 nodes |

### ****20. Explain how you can disable cowsay?****

If cowsay is installed, then by executing playbooks inside the Ansible, you can disable cowsay by using the two ways given below:

* Uninstall cowsay
* Set up value for the environment variable

export ANSIBLE\_NOCOWS = 1

### ****21. What is Ansible-doc?****

Ansible-doc displays information on modules installed in Ansible libraries. It displays a listing of plug-ins and their short descriptions, provides a printout of their documentation strings, and creates a short snippet that can be pasted in a playbook.

### ****22. What is the code you need to write for accessing a variable name?****

The following command will do the job:

*{{ hostvars[inventory\_hostname]['ansible\_' + which\_interface]['ipv4']['address'] }}*

The method of using hostvars is important because it is a dictionary of the entire namespace of variables. ‘inventory\_hostname’ variable specifies the current host you are looking over in the host loop.

### ****23. What is the method to check the inventory vars defined for the host?****

This can be done by using the following command:

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

### ****24. Explain Ansible facts****

Ansible facts can be thought of as a way for Ansible to get information about a host and store it in variables for easy access. This information stored in predefined variables is available to use in the playbook. To generate facts, Ansible runs the set-up module.

### ****25. When should you test playbooks and roles?****

In Ansible, tests can be added either in new playbooks or to existing playbooks. Therefore, most testing jobs offer clean hosting each time we use them. By using this testing methodology, we need to make very minute or zero code changes.

### ****26. Discuss the method to create an empty file with Ansible****

To create an empty file you need to follow the steps given below:

* **Step 1:** Save an empty file into the files directory
* **Step 2:** Copy it to the remote host

### ****27. Explain Ansible modules in detail****

Ansible modules are small pieces of code that perform a specific task. Modules can be used to automate a wide range of tasks. Ansible modules are like functions or standalone scripts that run specific tasks idempotently. Their return value is JSON strings in stdout and its input depends on the type of module.

There are two types of modules:

* **Core modules:** These are modules that the core Ansible team maintains and will always ship with Ansible itself. The issues reported are fixed on priority than those in the extras repo. The source of these modules is hosted by Ansible on GitHub in Ansible-modules-core.
* **Extras Modules:** The Ansible community maintains these modules; so, for now, these are being shipped with Ansible but they might get discontinued in the future. Popular extras modules may be promoted to core modules over time. The source for these modules is hosted by Ansible on GitHub in Ansible-modules-extras.

### ****28. What are callback plug-ins in Ansible?****

Callback plug-ins mostly control the output we see while running CMD programs. Apart from this, it can also be used for adding additional output or multiple outputs. For example, log\_plays callback is used to record playbook events into a log file and mail callback is used to send an email on playbook failures.

You can also add custom callback plug-ins by dropping them into a callback\_plugins directory adjacent to play, inside a role, or by putting it in one of the callback directory sources configured in ansible.cfg.

### ****29. What is Ansible inventory and its types?****

An Ansible inventory file is used to define hosts and groups of hosts upon which the tasks, commands, and modules in a playbook will operate.

In Ansible, there are two types of inventory files, static and dynamic.

* **Static inventory:** Static inventory file is a list of managed hosts declared under a host group using either hostnames or IP addresses in a plain text file. The managed host entries are listed below the group name in each line.
* **Dynamic inventory:** Dynamic inventory is generated by a script written in Python or any other programming language or, preferably, by using plug-ins. In a cloud set-up, static inventory file configuration will fail since IP addresses change once a virtual server is stopped and started again.

***Check out our***[***DevOps training course***](https://intellipaat.com/devops-certification-training/)***especially curated by industry experts!***

### ****30. What is an Ansible vault?****

Ansible vault is used to keep sensitive data, such as passwords, instead of placing it as plain text in playbooks or roles. Any structured data file or single value inside a YAML file can be encrypted by Ansible.

**To encrypt the data:**

**Command:** *ansible-vault encrypt foo.yml bar.yml baz.yml*

**To decrypt the data:**

***Command:*** *ansible-vault decrypt foo.yml bar.yml baz.yml*

### ****31. How do we write an Ansible handler with multiple tasks?****

Suppose you want to create a handler that restarts a service only if it is already running.

Handlers can understand generic topics, and tasks can notify those topics as shown below. 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*

### ****32. How to generate encrypted passwords for a user module?****

We can do this by using a small code:

*ansible all -i localhost, -m debug -a "msg={{ 'mypassword' | password\_hash('sha512', 'mysecretsalt') }}"*

We can also use the Passlib library of Python.

**Command:***python -c "from passlib.hash import sha512\_crypt; import getpass; print(sha512\_crypt.using(rounds=5000).hash(getpass.getpass()))"*

### ****33. Explain the concept of blocks under Ansible?****

Blocks allow for logical grouping of tasks and in-play error-handling. Most of what you can apply to a single task can be applied at the block level, which also makes it much easier to set data or directives common to the tasks. This does not mean that the directive affects the block itself but is inherited by the tasks enclosed by a block, i.e., it will be applied to the tasks, not the block itself.

### ****34. Do you have any idea of how to turn off the facts in Ansible?****

If you do not need any factual data about the hosts and know everything about the systems centrally, we can turn off fact gathering. This has advantages in scaling Ansible in push mode with very large numbers of systems, mainly, or if we are using Ansible on experimental platforms.

**Command:**

- hosts: whatever

gather\_facts: no

### ****35. What are the registered variables under Ansible?****

Registered variables are valid on the host for the remainder of the playbook run, which is the same as the lifetime of facts in Ansible. Effectively registered variables are very similar to facts. While using register with a loop, the data structure placed in the variable during the loop will contain a results attribute, which is a list of all responses from the module.

### ****36. By default, the Ansible reboot module waits for how many seconds. Is there any way to increase it?****

By default, the Ansible reboot module waits 600 seconds. Yes, it is possible to increase Ansible reboot to certain values. The syntax given-below can be used for the same:

*- name: Reboot a Linux system*

*reboot:*

*reboot\_timeout: 1200*

### ****37. What do you understand by the term idempotency?****

Idempotency is an important Ansible feature. It prevents unnecessary changes in managed hosts. With idempotency, we can execute one or more tasks on a server as many times as we need to, but it will not change anything that has already been modified and is working correctly.

To put it simply, the only changes added are the ones needed and not already in place.

### ****38. Can you copy files recursively onto a target host? If yes, how?****

We can copy files recursively onto a target host by using the copy module. It has a recursive parameter that copies files from a directory. There is another module called synchronize, which is specifically made for this.

*- synchronize:*

*src: /first/absolute/path*

*dest: /second/absolute/path*

*delegate\_to: "{{ inventory\_hostname }}"*

### ****39. Can you keep data secret in the playbook?****

The following playbook might come in handy if you want to keep secret any task in the playbook when using -v (verbose) mode:

*- name: secret task*

*shell: /usr/bin/do\_something --value={{ secret\_value }}*

*no\_log: True*

### **1. Let’s begin with the basics. What is Ansible?**

Ansible is an open-source platform that facilitates configuration management, task automation, or application deployment. It is a valuable DevOps tool. It was written in [Python](https://www.simplilearn.com/learn-the-basics-of-python-article) and powered by Red Hat. It uses SSH to deploy SSH without incurring any downtime.

2. List Ansible’s advantages

Ansible has many strengths, including:

* It’s agentless and only requires SSH service running on the target machines
* Python is the only required dependency and, fortunately, most systems come with the language pre-installed
* It requires minimal resources, so there’s low overhead
* It’s easy to learn and understand since Ansible tasks are written in YAML.
* Unlike other tools, most of which are Procedural, ansible is declarative; define the desired state, and Ansible fulfills the requirements needed to achieve it

3. What are CD and CI, and what is Ansible’s relationship with them?

CD stands for continuous delivery, and CI stands for[continuous integration;](https://www.simplilearn.com/tutorials/devops-tutorial/continuous-integration) both are software development practices.

In CD, developers build software that can be released into production at any given time. CI, on the other hand, consists of each developer uploading regularly scheduled integrations (usually daily), resulting in multiple integrations every day. Ansible is an ideal tool for [CI/CD processes](https://www.simplilearn.com/tutorials/devops-tutorial/continuous-delivery-and-continuous-deployment), providing a stable infrastructure for provisioning the target environment and then deploying the application to it.

### **4. Describe how Ansible works.**

This is one of the most frequently asked ansible interview questions where the interviewer wants to know whether you actually know the tool in and out or not. You can start this way - ansible is broken down into two types of servers: controlling machines and nodes. [Ansible is installed](https://www.simplilearn.com/tutorials/ansible-tutorial/ansible-installation) on the controlling computer, and the controlling machines manage the nodes via SSH.

The controlling machine contains an inventory file that holds the node system’s location. Ansible runs the playbook on the controlling machine to deploy the modules on the node systems. Since Ansible is agentless, there’s no need for a third-party tool to connect the nodes.

### **5. State the requirements for the Ansible server.**

You need a virtual machine with Linux installed on it, running with Python version 2.6 or higher.

### **6. Explain what a “playbook” is.**

A playbook has a series of YAML-based files that send commands to remote computers via scripts. Developers can configure entire complex environments by passing a script to the required systems rather than using individual commands to configure computers from the command line remotely. Playbooks are one of Ansible’s strongest selling points and often referred to as the tool’s building blocks.

### **7. How do you set up Ansible?**

You can use either the Python installer or a Linux-based installation process, such as apt or yum.

### **8. What is Ansible Tower?**

It’s an enterprise-level web-based solution that increases Ansible’s accessibility to other [IT teams](https://www.simplilearn.com/areas-to-focus-while-upskilling-your-it-teams-article) by including an easy-to-use UI (user interface). Tower’s primary function is to serve as the hub for all of an organization’s automation tasks, allowing users to monitor configurations and conduct rapid deployments.

### **9. What is “idempotency”?**

idempotency is an important Ansible feature. It prevents unnecessary changes in the managed hosts. With idempotency, you can execute one or more tasks on a server as many times as you need to, but it won’t change anything that’s already been modified and is working correctly. To put it in basic terms, the only changes added are the ones needed and not already in place.

### **10. What is Ansible Galaxy?**

This is a tool bundled with Ansible to create a base directory structure. Galaxy is a website that lets users find and share Ansible content. You can use this command to download roles from the website:

$ ansible-galaxy install username.role\_name

### **11. How do you use Ansible to create encrypted files?**

To create an encrypted file, use the ‘ansible-vault create’ command.

$ ansible-vault create filename.yaml

You will get a prompt to create a password, and then to type it again for confirmation. You will now have access to a new file, where you can add and edit data.

### **12. What are “facts” in the context of Ansible?**

Facts are newly discovered and known system variables, found in the playbooks, used mostly for implementing conditionals executions. Additionally, they gather ad-hoc system information.

You can get all the facts by using this command:

$ ansible all- m setup

### **13. Explain what an ask\_pass module is.**

It’s a playbook control module used to control a password prompt. It’s set to True by default.

### **14. What’s an ad hoc command?**

Users initiate ad hoc commands to initiate actions on a host without using a playbook. Consider it a one-shot command.

### **15. Explain the difference between a playbook and a play.**

A play is a set of tasks that run on one or more managed hosts. Plays consist of one or more tasks. A playbook consists of one or more plays.

### **16. What exactly is a configuration management tool?**

[Configuration management tools](https://www.simplilearn.com/configuration-management-2-article) help keep a system running within the desired parameters. They help reduce deployment time and substantially reduce the effort required to perform repetitive tasks. Popular configuration management tools on the market today include [Chef, Puppet](https://www.simplilearn.com/chef-vs-puppet-differences-and-similarities-article), Salt, and of course, Ansible.

Finally, let us go through the Ansible interview questions at an advanced level.

## **Advanced Ansible Interview Questions For Experienced Professionals**

### **17. What are tags?**

When there’s an extensive playbook involved, sometimes it’s more expedient to run just a part of it as opposed to the entire thing. That’s what tags are for.

### **18. Speaking of tags, how do you filter out tasks?**

You can filter out tasks in one of two ways:

* Use –tags or –skip-tags options on the command line
* If you’re in Ansible configuration settings, use the TAGS\_RUN and TAGS\_SKIP options.

### **19. What’s a handler?**

In Ansible, a handler is similar to a regular task in a playbook, but it will only run if a task alerts the handler. Handlers are automatically loaded by roles/<role\_name>/handlers/main.yaml. Handlers will run once, after all of the tasks are completed in a particular play.

### **20. How do you test Ansible projects?**

This is another frequently asked ansible interview question. Try elaborating the answer to this question rather than just answering the testing methods in one word. There are three testing methods available:

#### **Asserts**

Asserts duplicates how the test runs in other languages like Python. It verifies that your system has reached the actual intended state, not just as a simulation that you’d find in check mode. Asserts shows that the task did the job it was supposed to do and changed the appropriate resources.

#### **Check Mode**

Check mode shows you how everything would run if no simulation was done. Therefore, you can easily see if the project behaves the way you want it to. On the downside, check mode doesn’t run scripts and commands used in roles and playbooks. To get around this, you have to disable check mode for specific tasks by running “check\_mode: no.”

#### **Manual Run**

Just run the play and verify that the system is in its desired state. This testing choice is the easiest method, but it carries an increased risk because the results in a test environment may not be the same in a production environment.

### **21. How do you upgrade Ansible?**

Upgrading Ansible is easy. Just use this command: sudo pip install ansible==<version-number>

### **22. When do you use {{ }}?**

One of Ansible’s most basic rules is: “Always use {{ }} except when:”

### **23. Explain how to access shell environment variables.**

You can access the controlling machine’s existing variables by using the “env” lookup plugin. For instance, to access the value of the management machine’s home environment variable, you’d enter:

local\_home:”{{lookup(‘env’,’HOME’)}}”

### **24. How do you keep data secret in a playbook?**

If you want to keep secret data but still be able to share it publicly, then use Vault in playbooks. But if you’re using –v (verbose) mode and don’t want anyone to see the results, then use:

name: secret task

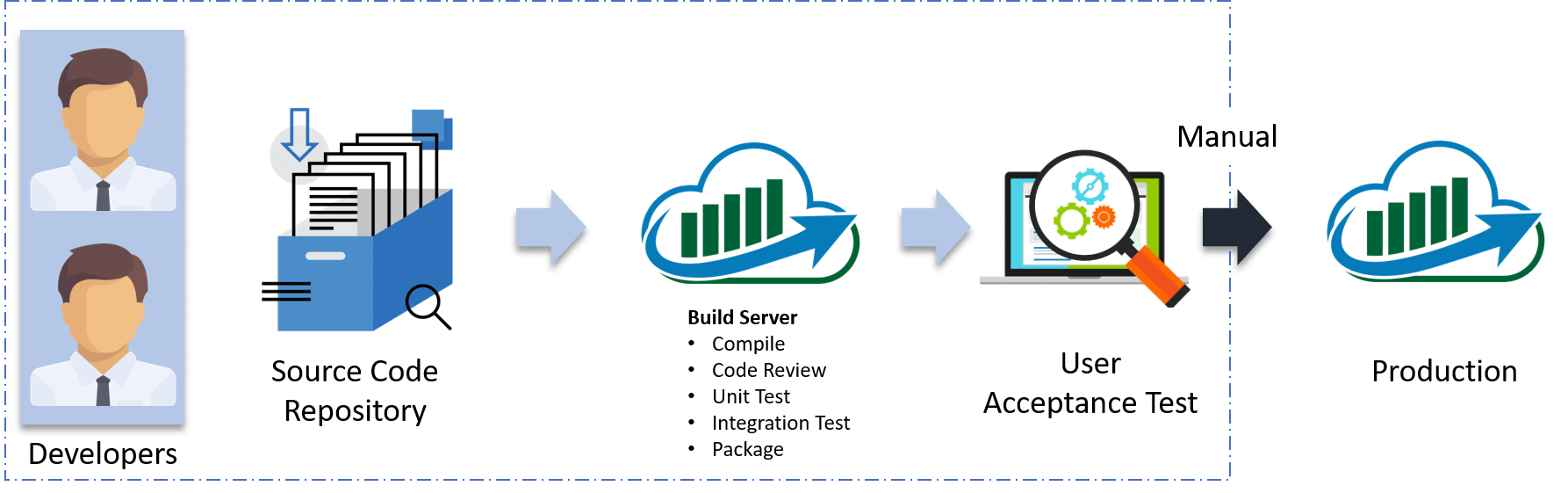
shell: /usr/bin/do\_something --value={{ secret\_value }}

no\_log: True

### ****Q1. What is CI/CD?****

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually, each person integrates at least daily leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.

Continuous Delivery is a process where you build software in such a way that it can be released to production at any time. Consider the diagram below:



Let me explain the above diagram:

* Automated build scripts will detect changes in Source Code Management (SCM) like Git.
* Once the change is detected, source code would be deployed to a dedicated build server to make sure build is not failing and all test classes and integration tests are running fine.
* Then, the build application is deployed on the test servers (pre-production servers) for User Acceptance Test (UAT).
* Finally, the application is manually deployed on the production servers for release.

### ****Q2. What is Configuration Management and how does it help an organization?****

Configuration Management is the practice of handling updates and changes systematically so that a system maintains its integrity over time. Configuration Management (CM) keeps a track of all the updates that are needed in a system and it ensures that the current design and build state of the system is up to date and functioning correctly.

Configuration Management can help an organization by overcoming the following challenges:

* Finding out what changes need to be implemented when user requirements change.
* Redoing and updating an implementation due to change in the requirements since the last implementation.
* Reverting to an older version of the component because the latest version is flawed.
* Replacing the wrong component because you couldn’t accurately determine which component needed replacing.

To better understand this consider the NYSE example:

The New York Stock Exchange (NYSE) encountered a glitch in their software which prevented them from trading stocks for approx 90 minutes. On the night before, new software was installed on 8 of its 20 trading terminals. Unfortunately, the software failed to operate properly on the 8 terminals.

Therefore, by using Configuration Management tools such as Ansible and Puppet, they reverted back to the old software. Had they not implemented CM, they would’ve taken a much longer time to fix the issue which would lead to a much bigger loss.

### ****Q3. What is Ansible and what makes it stand out from the rest of the Configuration Management tools?****

[*Ansible*](https://www.edureka.co/blog/what-is-ansible/) is an open source IT Configuration Management, Deployment & Orchestration tool. It aims to provide large productivity gains to a wide variety of automation challenges.

Here’s a list of features that makes Ansible such an effective Configuration Management and Automation tool:

1. Simple: Uses a simple syntax written in YAML called playbooks.
2. Agentless: No agents/software or additional firewall ports that you need to install on the client systems or hosts which you want to automate.
3. Powerful and Flexible: Ansible’s capabilities allow you to orchestrate the entire application environment regardless of where it is deployed.
4. Efficient: Ansible introduces modules as basic building blocks for your software. So, you can even customize it as per your needs.

### ****Q4. How does Ansible work?****

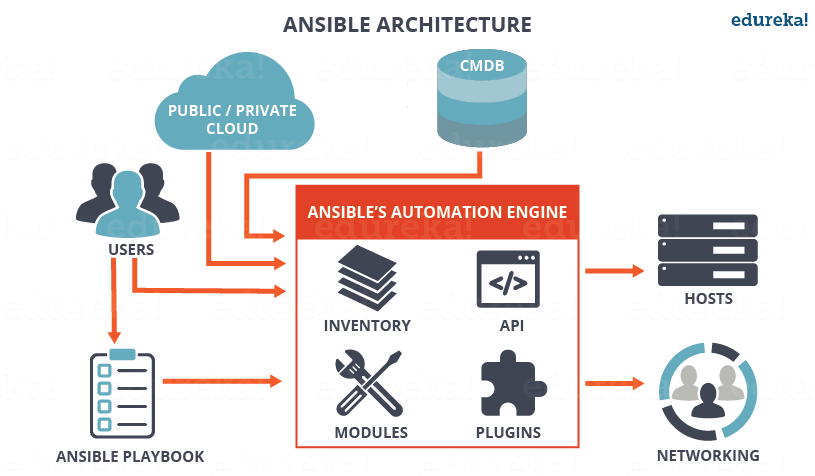
Ansible, unlike other configuration management tools, is categorized into two types of servers – Controlling machines and Nodes. Controlling machine is where Ansible is installed and nodes are the ones that are managed by the controlling machines through SSH. There is an inventory file in the controlling machine that holds the location of the node systems. Ansible deploys modules on the node systems by running the playbook on the controlling machine. Ansible is agentless, that means there is no need to have a third party tool to make a connection between one node and the other.

### ****Q5. How is Ansible different from Puppet?****

|  |  |  |
| --- | --- | --- |
| **Metrics** | **Ansible** | **Puppet** |
| 1. Availability 2. Ease of set up 3. Management 4. Scalability 5. Configuration language 6. Interoperability 7. Pricing nodes | * Highly available * Easy * Easy management * Highly scalable * YAML(Python) * High * $10,000 | * Highly available * Comparatively hard to set up * Not very easy * Highly scalable * DSL(PuppetDSL) * High * $11200-$19900 |

### ****Q6. What are the different components of ansible? Explain Ansible architecture.****

The below diagram depicts the Ansible architecture:



Ansible Architecture – Ansible Interview Questions – Edureka

The main component of Ansible is the Ansible automation engine. This engine directly interacts with various cloud services, Configuration Management Database (CMBD) and different users who write various playbooks to execute the Ansible Automation engine.

The Ansible Automation engine consists of the following components:

**Inventories:**These are a list of nodes containing their respective IP addresses, servers, databases, etc. which needs to be managed.

**APIs:** Just like any other API, the Ansible APIs are used for commuting various Cloud services, public or private services.

**Modules:** The modules are used to manage system resources, packages, libraries, files, etc. Ansible modules can be used to automate a wide range of tasks. Ansible provides around 450 modules that automate nearly every part of your environment.

**Plugins:** If you want to execute Ansible tasks as a job, Ansible Plugins can be used. They simplify the execution of a task by building a job like an environment that basically contains pieces of code corresponding to some specific functionality. There are 100s of Plugins provided by Ansible. An example is the Action plugin, which acts as front ends to modules and can execute tasks on the controller before calling the modules themselves.

**Networking:** Ansible can also be used to automate different networks and services. It can do this by creating a playbook or an Ansible role that easily spans different network hardware.

**Hosts:** The Ansible Hosts/ Node systems are machines (Linux, Windows, etc) that are getting automated.

**Playbooks:** Playbooks are simple code files which describe the tasks that need to be executed. The Playbooks are written in YAML format. They can be used to automate tasks, declare configurations, etc.

**CMDB:** It is a database that acts as a storehouse for various IT installations. It holds data about various IT assets (also known as configuration items (CI)) and describes the relationships between such assets.

**Cloud:** It is a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server.

To learn more about Ansible, you can go through this [Ansible Tutorial](https://www.edureka.co/blog/ansible-tutorial/) blog.

### ****Q7. What are Ansible Server requirements?****

If you are a windows user then you need to have a virtual machine in which Linux should be installed. It requires Python 2.6 version or higher. you fulfill these requirements and you’re good to go!

### ****Q8. How would you install Ansible on a CentOS system?****

This can be done in two simple steps:

**Step 1:** Set up EPEL Repository

EPEL (Extra Packages for Enterprise Linux) is an open source and free community-based repository project from Fedora team which provides high-quality add-on software packages for Linux distribution including RHEL (Red Hat Enterprise Linux), CentOS, and Scientific Linux.

The Ansible package is not available in the default yum repositories, so we will enable EPEL repository by using the below command:

sudo rpm -ivh http://dl.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm

This will download all the necessary packages which will be required to install Ansible.

**Step 2:** Install Ansible

Now that your EPEL repository has been added, all you have to do now is install Ansible using the command below:

yum install ansible -y

That’s all! It’s a two-step process that barely takes a minute!

If you wish to check the version of Ansible installed on your system, use the command below:

ansible –version

**Q9. Explain a few of the basic terminologies or concepts in Ansible.**

Few of the basic terms that are commonly used while operating on Ansible are:

**Controller Machine:** The Controller machine is responsible for provisioning the servers that are being managed. It is the machine where Ansible is installed.

**Inventory:** An inventory is an initialization file that has details about the different servers you are managing.

**Playbook:** It is a code file written in the YAML format. A playbook basically contains the tasks that need to be executed or automated.

**Task:** Each task represents a single procedure that needs to be executed, e.g. Install a library.

**Module:** A module is a set of tasks that can be executed. Ansible has 100s of built-in modules, but you can also create custom ones.

**Role:** An Ansible role is a pre-defined way for organizing playbooks and other files in order to facilitate sharing and reusing portions of provisioning.

**Play:** A task executed from start to finish or the execution of a playbook is called a play.

**Facts:** Facts are global variables that store details about the system, like network interfaces or operating system.

**Handlers:** Are used to trigger the status of a service, such as restarting or stopping a service.

### ****Q10. Explain the concept behind Infrastructure as Code (IaC).****

Infrastructure as Code (IaC) is a process for managing and operating data servers, storage systems, system configurations, and network infrastructure.

In traditional configuration management practices, each minute configuration change required manual action by system administrators and the IT support team. But with IaC, all the configuration details are managed and stored in a standardized file system, wherein the system automatically manages infrastructure changes and deals with system configurations.

Therefore, we do not require most of the manual effort since everything is managed and automated by following the IaC approach. Tools such as Ansible can be used to implement IaC approach.

### ****Q11. Compare Ansible with Chef.****

|  |  |  |
| --- | --- | --- |
| **Metrics** | **Ansible** | **Chef** |
| 1. Availability 2. Ease of set up 3. Management 4. Scalability 5. Configuration language 6. Interoperability 7. Pricing nodes | * Highly available * Easy * Easy management * Highly scalable * YAML(Python) * High * $10,000 | * Highly available * Not very easy * Not very easy * Highly scalable * DSL(Ruby) * High * $13700 |

### ****Q12. What is Ansible Galaxy?****

Galaxy is a website that lets Ansible users share their roles and modules. The Ansible Galaxy command line tool comes packed with Ansible, and it can be used to install roles from Galaxy or directly from a Source Control Management system such as Git. It can also be used to build new roles, remove existing ones and perform tasks on the Galaxy website.

You can use the below command to download roles from the Galaxy website:

$ansible-galaxy install username.role\_name

### ****Q13. What are Ad-hoc commands? Give an example.****

Ad-hoc commands are simple one-line commands used to perform a certain task. You can think of Adhoc commands as an alternative to writing playbooks. An example of an Adhoc command is as follows:

ansible host -m netscaler -a "nsc\_host=nsc.example.com user=apiuser password=apipass"

[[](https://www.edureka.co/devops-certification-training)](https://www.edureka.co/devops-certification-training" \t "_blank)

### [DevOps Certification Training Course](https://www.edureka.co/devops-certification-training" \t "_blank)

[Explore Curriculum](https://www.edureka.co/devops-certification-training" \t "_blank)

The above Adhoc command accesses the netscaler module to disable the server.

### ****Q14. What are the variables in Ansible?****

Variables in Ansible are very similar to variables in any programming language. Just like any other variable, an Ansible variable is assigned a value which is used in computing playbooks. You can also use conditions around the variables. Here’s an example:

|  |  |
| --- | --- |
| 1  2  3 | - hosts: your hosts  vars:  port\_Tomcat : 8080 |

Here, we’ve defined a variable called port\_Tomcat and assigned the port number 8080 to it. Such a variable can be used in the Ansible Playbook.

### ****Q15. What is the difference between a variable name and an environment variable?****

|  |  |
| --- | --- |
| Variable name | Environment variable |
| * You need to add strings to create variable names * You can easily create multiple variable names by adding strings * We use the ipv4 address for variable names | * You need existing variables to access environment variables. * To create environment variables we must refer advanced Ansible playbook * We use {{ ansible\_env.SOME\_VARIABLE }} for remote environment variables. |

### ****Q16. What are the Ansible Modules? Explain the different types.****

Ansible modules are a small set of programs that perform a specific task. Modules can be used to automate a wide range of tasks. Modules in Ansible are considered to be idempotent or in other words, making multiple identical requests has the same effect as making a single request.

There are 2 types of modules in Ansible:

1. Core modules
2. Extras modules

**Core Modules**

These are modules that the core Ansible team maintains and will always ship with Ansible itself. They will also receive a slightly higher priority for all requests than those in the “extras” repos. The source of these modules is hosted by Ansible on GitHub in the Ansible-modules-core.

**Extras Modules**

These modules are currently shipped with Ansible but might be shipped separately in the future. They are also mostly maintained by the Ansible Community. Non-core modules are still fully usable but may receive slightly lower response rates for issues and pull requests.

Popular “extras” modules may be promoted to core modules over time. The source for these modules is hosted by Ansible on GitHub in the Ansible-modules-extras.

### ****Q17. What is Ansible Task?****

Ansible Tasks allow you to break up bits of configuration policy into smaller files. These are blocks of code that can be used to automate any process. For example, if you wish to install a package or update a software, you can follow the below code snippet:

Install <package\_name>, update <software\_name>

### ****Q18. Can you explain what are playbooks in Ansible? Explain with some examples.****

Playbooks in Ansible are written in the YAML format. It is a human-readable data serialization language. It is commonly used for configuration files. It can also be used in many applications where data is being stored.

For Ansible, nearly every YAML file starts with a list. Each item in the list is a list of key/value pairs, commonly called a “hash” or a “dictionary”. So, we need to know how to write lists and dictionaries in YAML.

All members of a list are lines beginning at the same indentation level starting with a “- ” (dash and space). More complicated data structures are possible, such as lists of dictionaries or mixed dictionaries whose values are lists or a mix of both.

For example, if you want a playbook containing details about the USA:

|  |  |
| --- | --- |
| 1  2  3  4 | -USA  -continent: North America  -capital: Washington DC<a name="AnsibleIntermediateLevelInterviewQuestions"></a>  -population: 319 million |

Now that you know the basic level questions, let’s take a look at a little more advanced Ansible Interview Questions.

## **Intermediate Level Ansible Interview Questions**

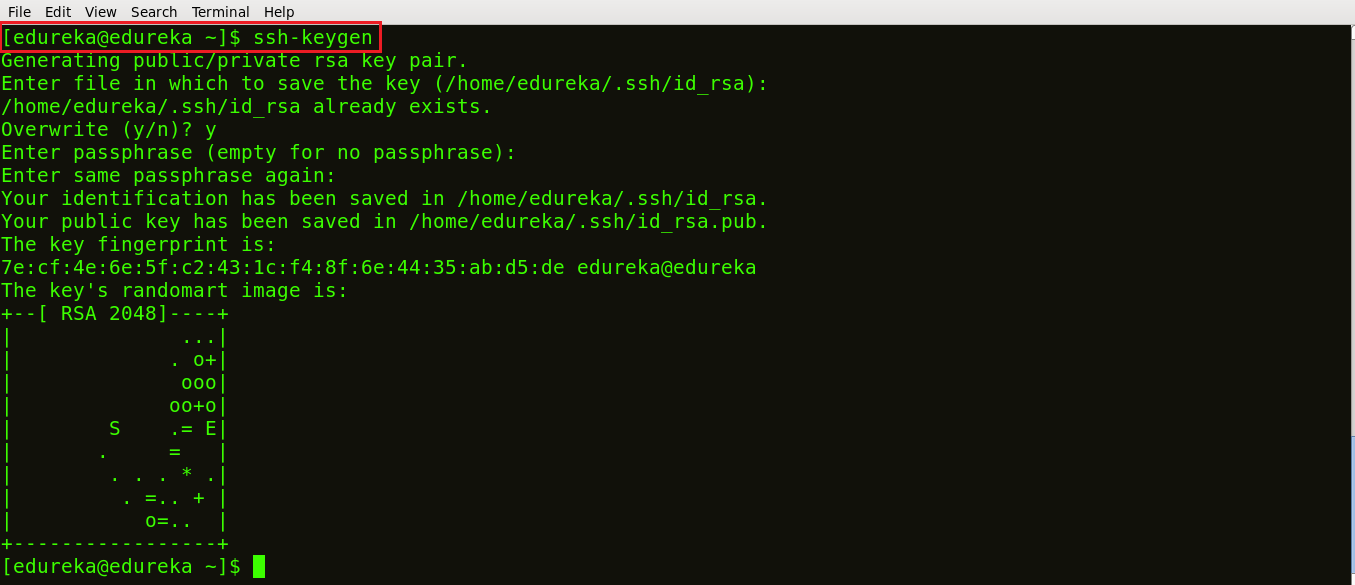
Once you’ve aced the basic conceptual questions, the interviewer will increase the difficulty level. So let’s move on to the next section of this Ansible Interview Questions article. This section talks about the commands that are very common amongst docker users.

### ****Q19. Can you write a simple playbook to install Nginx on a host machine?****

**Step 1:** Generate a public SSH key and by using SSH connect to your host.

Follow the command below:

$ ssh-keygen

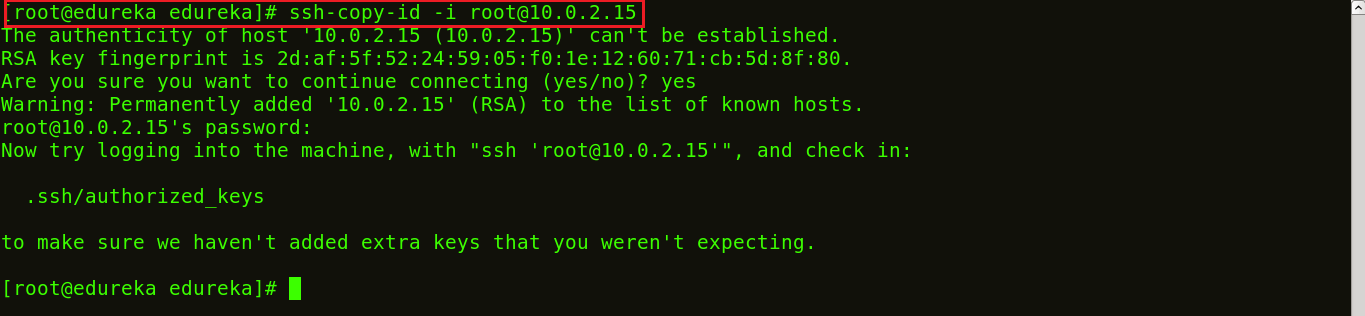


Ansible Playbook – Ansible Interview Questions – Edureka

As shown above, a public SSH key is generated.

**Step 2:** Next, copy the public SSH key on your hosts. Follow the below command to do it:

ssh-copy-id -i root@IP address of your host

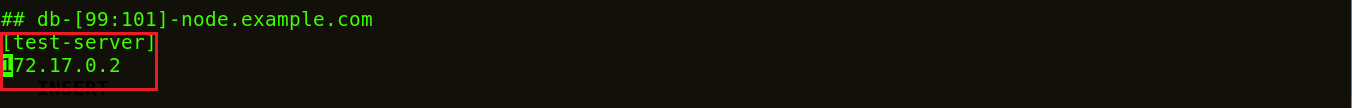


Ansible Playbook – Ansible Interview Questions – Edureka

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

Follow the below command:

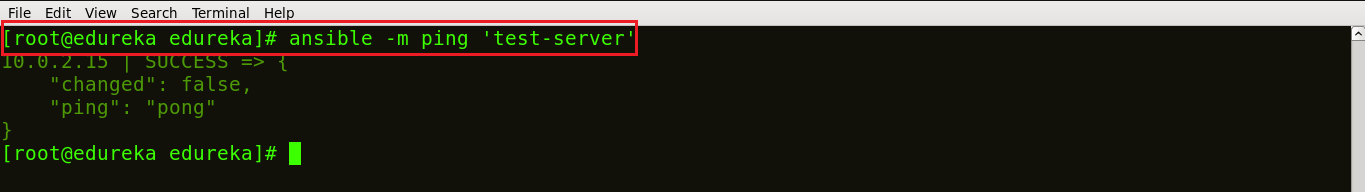
vi /etc/ansible/hosts



Ansible Playbook – Ansible Interview Questions – Edureka

Once you run the command, the vi editor will open where you can list down the IP addresses of your hosts. This is now your inventory.

**Step 4:** To check if the connection has been established, let’s ping:

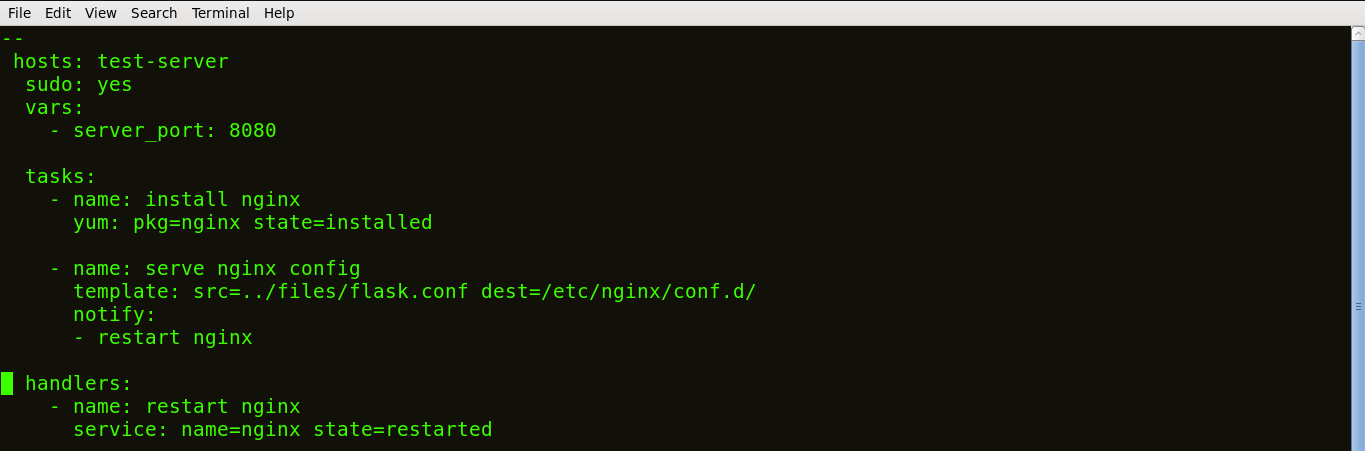


Ansible Playbook – Ansible Interview Questions – Edureka

The above image shows that a connection has been made between the control machine and the host.

**Step 5:** Create a playbook to install Nginx on the host machine.  To create a playbook you just need to open a file with a yml extension, like shown below:

vi <Name of your file>.yml



Ansible Playbook – Ansible Interview Questions – Edureka

In an Ansible Playbook, the tasks are defined as a list of dictionaries and are executed from top to bottom.

Each task is defined as a dictionary that can have several keys, such as “name” or “sudo” which signify the name of the task and whether it requires sudo privileges.

A variable server\_port is set that listens on TCP port 8080 for incoming requests.

Here, the first task is to get the necessary package for installation of Nginx and then install it. Internally, Ansible will check if the directory exists and create it if it’s not, otherwise, it will do nothing.

The next task is to configure Nginx. In Nginx, contexts contain configuration details.

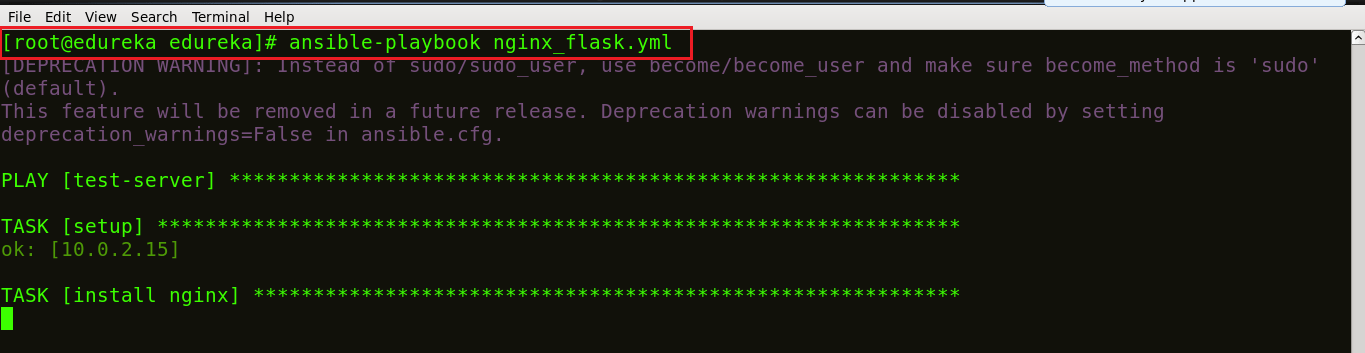
Here, the template is a file you can deploy on hosts. However, template files also include some reference variables which are pulled from variables defined as part of an Ansible playbook or facts gathered from the hosts. Facts containing the configuration details are being pulled from a source directory and being copied to a destination directory.

Handlers here define the action to be performed only upon notification of tasks or state changes. In this playbook, we defined, notify: restart Nginx handler which will restart Nginx once the files and templates are copied to hosts.

Now, save the file and exit.

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

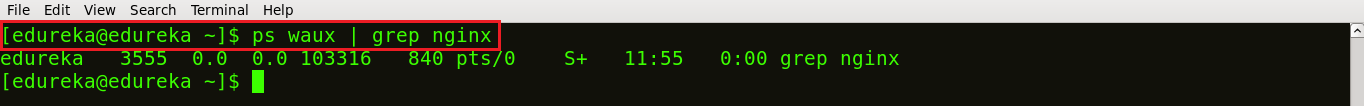
ansible-playbook <name of your file>.yml



Ansible Playbook – Ansible Interview Questions – Edureka

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

ps waux | grep nginx



Ansible Playbook – Ansible Interview Questions – Edureka

In the above image, the different process IDs 3555 and 103316 are running which shows that Nginx is running on your host machines.

### ****Q20. How would you access a variable of the first host in a group?****

This can be done by executing the below command:

{{ hostvars[groups['webservers'][0]]['ansible\_eth0']['ipv4']['address'] }}

In the above command, we’re basically accessing the hostname of the first machine in the webservers group. If you’re using a template to do this, use the Jinja2 ‘#set’ or you can also use set\_fact, like shown below:

|  |  |
| --- | --- |
| 1  2 | - set\_fact: headnode={{ groups[['webservers'][0]] }}  - debug: msg={{ hostvars[headnode].ansible\_eth0.ipv4.address }} |

### ****Q21. Why is ‘{{ }}’ notation used? And how can one interpolate variables or dynamic variable names?****

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.

### ****Q22. What is Ansible role and how are they different from the playbook?****

[Ansible Roles](https://www.edureka.co/blog/ansible-roles-setup-mean-stack)is basically another level of abstraction used to organize playbooks. They provide a skeleton for an independent and reusable collection of variables, tasks, templates, files, and modules which can be automatically loaded into the playbook. Playbooks are a collection of roles. Every role has specific functionality.

Let’s understand the difference between Ansible roles and playbook with an example.

Suppose you want your playbook to perform 10 different tasks on 5 different systems, would you use a single playbook for this? No, using a single playbook can make it confusing and prone to blunders. Instead, you can create 10 different roles, where each role will perform one task. Then, all you need to do is, mention the name of the role inside the playbook to call them.

### ****Q23. How do I write an Ansible handler with multiple tasks?****

Suppose you want to create a handler that restarts a service only if it is already running.

Handlers can “listen” to generic topics, and tasks can notify those topics as shown below. 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:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | - 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 |

### ****Q24. How to keep secret data in a playbook?****

Suppose you have a task that you don’t want to show the output or command given to it when using -v (verbose) mode, the following task can be used to do it:

|  |  |
| --- | --- |
| 1  2  3 | - name: secret task  shell: /usr/bin/do\_something --value={{ secret\_value }}  no\_log: True |

This can be used to keep verbose output but hide sensitive information from others who would otherwise like to be able to see the output.

The no\_log attribute can also apply to an entire play:

|  |  |
| --- | --- |
| 1  2 | - hosts: all  no\_log: True |

### ****Q25. What are Ansible Vaults and why are they used?****

[Ansible Vault](https://www.edureka.co/blog/ansible-vault-secure-secrets) is a feature that allows you to keep all your secrets safe. It can encrypt entire files, entire YAML playbooks or even a few variables. It provides a facility where you can not only encrypt sensitive data but also integrate them into your playbooks.

Vault is implemented with file-level granularity where the files are either entirely encrypted or entirely unencrypted. It uses the same password for encrypting as well as for decrypting files which makes using Ansible Vault very user-friendly.

### ****Q26. How to create encrypted files using Ansible?****

To create an encrypted file, use the ‘ansible-vault create’ command and pass the filename.

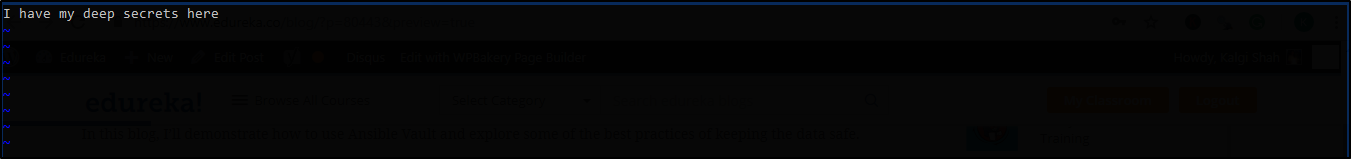
$ ansible-vault create filename.yaml

You’ll be prompted to create a password and then confirm it by re-typing it.

ansible vault create - Ansible Vault - Edureka

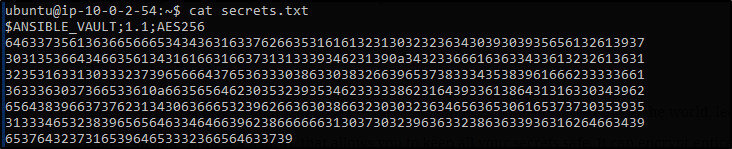
Ansible Playbook – Ansible Interview Questions – Edureka

Once your password is confirmed, a new file will be created and will open an editing window. By default, the editor for Ansible Vault is vi. You can add data, save and exit.



Ansible Playbook – Ansible Interview Questions – Edureka

This is your encrypted file:



Ansible Playbook – Ansible Interview Questions – Edureka

### ****Q27. What is Ansible Tower?****

[Ansible Tower](https://www.edureka.co/blog/ansible-tower) is Ansible at a more enterprise level. It is a web-based solution for managing your organization with a very easy user interface that provides a dashboard with all of the state summaries of all the hosts, allows quick deployments, and monitors all configurations.

The tower allows you to share the SSH credentials without exposing them, logs all the jobs, manage inventories graphically and syncs them with a wide variety of cloud providers.

### ****Q28. What features does the Ansible Tower provide?****

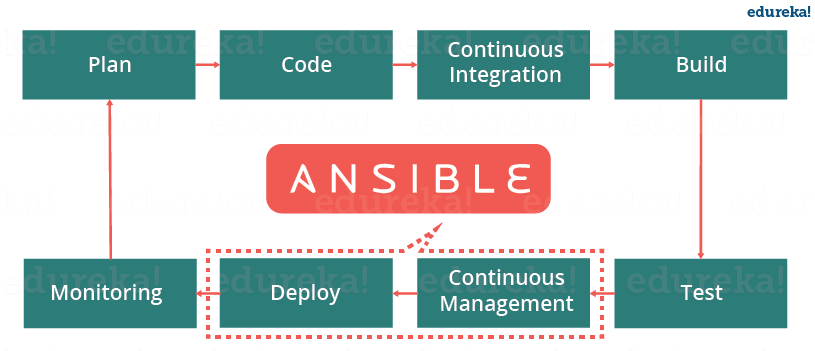
* **Ansible Tower Dashboard** – The Ansible Tower dashboard displays everything going on in your Ansible environment like the hosts, inventory status, the recent job activity and so on.
* **Real-Time Job Updates** – As Ansible can automate the complete infrastructure, you can see real-time job updates, like plays and tasks broken down by each machine either been successful or a failure. So, with this, you can see the status of your automation, and know what’s next in the queue.
* **Multi-Playbook Workflows** – This feature allows you to chain any number of playbooks, regardless of the usage of different inventories, utilizes various credentials, or runs different users.
* **Who Ran What Job When** – As the name suggests, you can easily know who ran what job where and when as, all the automation activity is securely logged in Ansible Tower.
* **Scale Capacity With Clusters** – 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.
* **Schedule Ansible Jobs** – Different kinds of jobs such as Playbook runs, cloud inventory updates, and source control updates can be scheduled inside Ansible Tower to run according to the need.
* **Manage & Track Inventory** – Ansible Tower helps you manage your entire infrastructure by letting you easily pull inventory from public cloud providers such as Amazon Web Services, Microsoft Azure, and more.
* **Self-Service** – This feature of Ansible Tower lets you launch Playbooks with just a single click. It can also, let you choose from available secure credentials or prompt you for variables and monitor the resulting deployments.
* **REST API & Tower CLI Tool** – Every feature present in Ansible Tower is available via Ansible Tower’s REST API, which provides the ideal API for a systems management infrastructure. The Ansible Tower’s CLI tool is available for launching jobs from CI systems such as Jenkins, or when you need to integrate with other command-line tools.
* **Remote Command Execution** – You can run simple tasks such as add users, restart any malfunctioning service, reset passwords on any host or group of hosts in the inventory with Ansible Tower’s remote command execution.

The following section will cover only the technical based Ansible Interview Questions. These questions are mostly based on the practical implementation of Ansible.

## **Ansible Technical Interview Questions**

### ****Q29. How is Ansible used in a Continuous Delivery pipeline? Explain.****

It is well known that in DevOps development and operations work is integrated. This integration is very important for modern test-driven applications. Hence, Ansible integrates this by providing a stable environment to both development and operations resulting in a smooth delivery pipeline.



Ansible In DevOps – Ansible Interview Questions – Edureka

When developers begin to think of infrastructure as part of their application i.e as Infrastructure as code (IaC), stability and performance become normative. 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. This is where Ansible automation plays a major role and stands out among its peers.

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.

### ****Q30. How can you create a LAMP stack and deploy a webpage by using Ansible?****

Suppose you’re trying to deploy a website on 30 systems, every website deployment will require a base OS, web-server, Database, and PHP. We use ansible playbook to install these prerequisites on all 30 systems at once.

For this particular problem statement, you can use two virtual machines, one as a server where Ansible is installed and the other machine acts as the remote host. Also, I’ve created a simple static webpage saved in a folder index which has two files, index.html, and style.css.

In the below code I’ve created a single Ansible playbook to install Apache, MySql, and PHP:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | ---  # 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 6 main tasks, each task performs a specific function:

* The first task adds the repository required to install MySQL and PHP.
* The second task installs apache2, MySQL-server, PHP, and PHP-MySQL.
* The third and fourth task starts the Apache and MySQL service.
* The fifth task creates a target directory in the host machine and
* Finally, the sixth task executes the index.html file, it picks up the file from the server machine and copies it into the host machine.

To finally run this playbook you can use the following command:

$ ansible-playbook lamp.yml -K

### ****Q31. How do I set the PATH or any other environment variable for a task?****

The environment variables can be set by using the ‘environment’ keyword. It can be set for either a task or an entire playbook as well. Follow the below code snippet to see how:

|  |  |
| --- | --- |
| 1  2  3 | environment:  PATH: "{{ ansible\_env.PATH }}:/thingy/bin"  SOME: value |

### ****Q32. How can one generate encrypted passwords for the user module?****

There are a couple of ways to do this. The simplest way is to use the ad-hoc command:

ansible all -i localhost, -m debug -a "msg={{ 'mypassword' | password\_hash('sha512', 'mysecretsalt') }}"

Another way is to use the mkpasswd functionality available on Linux systems:

mkpasswd --method=sha-512

However, if you’re using a macOS then you can generate these passwords using Python. To do this you must first install the Passlib password hashing library:

pip install passlib

After installing it, the SHA512 password values can be generated in the following manner:

python -c "from passlib.hash import sha512\_crypt; import getpass; print(sha512\_crypt.using(rounds=5000).hash(getpass.getpass()))"

### ****Q33. How can looping be done over a list of hosts in a group, inside of a template?****

An easy way to do this is to iterate over a list of hosts inside of a host group, in order to fill a template configuration file with a list of servers. This can be done by accessing the “$groups” dictionary in your template, like so:

|  |  |
| --- | --- |
| 1  2  3 | {% for host in groups['db\_servers'] %}  {{ host }}  {% endfor %} |

In order to access facts about these hosts, like, the IP address of each hostname, you need to make sure that the facts have been populated. For instance, make sure you have a play that talks to db\_servers:

|  |  |
| --- | --- |
| 1  2  3 | - hosts: db\_servers  tasks:  - debug: msg="doesn't matter what you do, just that they were talked to previously." |

Now you can use the facts within your template, like so:

|  |  |
| --- | --- |
| 1  2  3 | {% for host in groups['db\_servers'] %}  {{ hostvars[host]['ansible\_eth0']['ipv4']['address'] }}  {% endfor %} |

### ****Q34. How can I display all the inventory vars defined for my host?****

In order to check the inventory vars resulting from what you’ve defined in the inventory, you can execute the below command:

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

This will list down all the inventory vars.

### ****Q35. How should one configure a jump host to access servers that I have no direct access to?****

The first step would be to set a ProxyCommand in the ansible\_ssh\_common\_args inventory variable. All arguments that are defined in this variable are added to the sftp/scp/ssh command line when connecting to the relevant host. Let’s look at an example, consider the below inventory group:

|  |  |
| --- | --- |
| 1  2  3 | [gatewayed]  foo ansible\_host=192.0.2.1  bar ansible\_host=192.0.2.2 |

Next, you can create group\_vars/gatewayed.yml containing the following:

ansible\_ssh\_common\_args: '-o ProxyCommand="ssh -W %h:%p -q user@gateway.example.com"'

Ansible will then append these arguments to the command line while trying to connect to any hosts in the group gatewayed.

**Q36. How can you handle different machines needing different user accounts or ports to log in with?**

The simplest way to do this is by setting inventory variables in the inventory file.

Let’s consider that these hosts have different usernames and ports:

|  |  |
| --- | --- |
| 1  2  3 | [webservers]  asdf.example.com ansible\_port=5000 ansible\_user=alice  jkl.example.com ansible\_port=5001 ansible\_user=bob |

Also, if you wish to, you can specify the connection type to be used:

|  |  |
| --- | --- |
| 1  2  3  4 | [testcluster]  localhost ansible\_connection=local  /path/to/chroot1 ansible\_connection=chroot  foo.example.com ansible\_connection=paramiko |

To make this more clear it is best to keep these in group variables or file them in a group\_vars/<group-name> file.

### ****Q37. Is it unsafe to bulk-set task arguments from a variable?****

To set all the arguments in a task you can use the dictionary-typed variable. Even though this is usually good for dynamic executions, it induces a security risk. Therefore, when this happens, Ansible issues a warning. For example, consider the below code:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | vars:  usermod\_args:  name: testuser  state: present  tasks:  - user: '{{ usermod\_args }}' |

This example is safe but creating similar tasks is risky because the parameters and values passed to usermod\_args could be overwritten by malicious values in the host facts on a compromised target machine.

### ****Q38. Suppose you’re using Ansible to configure the production environment and your playbook uses an encrypted file. Encrypted files prompt the user to enter passwords. But since Ansible is used for automation, can this process be automated?****

Yes, Ansible uses a feature called password file, where all the passwords to your encrypted files can be saved. So each time the user is asked for the password, he can simply make a call to the password file. The password is automatically read and entered by Ansible.

$ ansible-playbook launch.yml --vault-password-file ~/ .vault\_pass.txt

Having a separate script that specifies the passwords is also possible. You need to make sure the script file is executable and the password is printed to standard output for it to work without annoying errors.

[[](https://www.edureka.co/devops-certification-training)](https://www.edureka.co/devops-certification-training" \t "_blank)

### [DevOps Certification Training Course](https://www.edureka.co/devops-certification-training" \t "_blank)

[Weekday / Weekend BatchesSee Batch Details](https://www.edureka.co/devops-certification-training" \t "_blank)

$ ansible-playbook launch.yml --vault-password-file ~/ .vault\_pass.py

### ****Q39. Have you worked with Ansible before? Please share your experience.****

Be very honest here. If you have used ansible before then talk about your experience. Talk about the projects that required ansible. You can tell the interviewer about how Ansible has helped you in provisioning and configuration management. If you haven’t used Ansible before then just talk about any related tools that you’ve used. These related tools could be anything like Git, Jenkins, Puppet, Chef, Satltstack, etc.

Be very honest because they know if you’re lying.

### ****Q40. Is Ansible an Open Source tool?****

Yes, Ansible is open source. That means you take the modules and rewrite them. Ansible is an open-source automated engine that lets you automate apps.

### ****Q41. How can you connect other devices within Ansible?****

Once Ansible is installed on the controlling machines, an inventory file is created. This inventory file specifies the connection between other nodes. A connection can be made using a simple SSH. To check the connection to a different device, you can use the ping module.

ansible -m ping all

The above command checks the connection to all the nodes specified in the inventory file.

### ****Q42. Is it possible to build our modules in Ansible?****

Yes, we can create our own modules within Ansible. It’s an open-source tool which basically works on python. You can start creating your own modules. The only requirements would be to be amazingly good at programming.

### ****Q43. What does Fact mean in Ansible?****

When any new variable about the system has been discovered it’s considered to be a “fact” in the playbook. Facts are mainly used to implement conditional executions. It can also be used to get the ad-hoc information about the system.

You can get facts with the following command:

$ ansible all- m setup

So when you want to extract only a part of the information, you use the setup module to filter out only the needed information.

### ****Q44. What is the ask\_pass module in Ansible?****

Ask\_pass is the control module in an Ansible playbook. This controls the prompting of the password when the playbook is getting executed. By default, it’s always set to True. If you are using SSH keys for authentication purposes then you really don’t have to change this setting at all.

### ****Q45. Explain the callback plugin in Ansible?****

Callback plugins are enable adding new behaviors to Ansible when responding to events. By default, callback plugins control most of the output you see when running the command line program. It can also be used to add additional output, integrate with other tools, etc.

### ****Q46. Does Ansible support AWS?****

Ansible has hundreds of modules supporting AWS and some of them include:

* Autoscaling groups
* CloudFormation
* CloudTrail
* CloudWatch
* DynamoDB
* ElastiCache
* Elastic Cloud Compute (EC2)
* Identity Access Manager (IAM)
* Lambda
* Relational Database Service (RDS)
* Route53
* Security Groups
* Simple Storage Service (S3)
* Virtual Private Cloud (VPC)

### ****Q47. Does Ansible support hardware provisioning?****

Yes, Ansible can provision hardware. A lot of companies are still stuck on to massive data centers of hardware. There are a few requirements. You must set up some services before you go ahead. Some of them are – DHCP, PXE, TFTP, Operating System Media, Web Server, etc.

### ****Q48. Write an Ansible playbook to automate the starting of EC2 instance.****

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | ---   - name: Create an ec2 instance    hosts: web    gather\_facts: false      vars:        region: us-east-1        instance\_type: t2.micro        ami: ami-05ea7729e394412c8        keypair: priyajdm      tasks:        - name: Create an ec2 instance        ec2:           aws\_access\_key: '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'           aws\_secret\_key: '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'           key\_name: "{{ keypair }}"           group: launch-wizard-26           instance\_type: "{{ instance\_type }}"           image: "{{ ami }}"           wait: true           region: "{{ region }}"           count: 1           vpc\_subnet\_id: subnet-02f498e16fd56c277           assign\_public\_ip: yes      register: ec2 |

* We start by mentioning AWS access key id and secret key using the parameters **aws\_access\_key** and **aws-secret\_key**.
* **key\_name:**pass the variable that defines the keypair being used here
* **group:**mention the name of the security group. This defines the security rules of the EC2 instance we’re trying to bring up
* **instance\_type:**pass the variable that defines the type of instance we’re using here
* **image:**pass the variable that defines the AMI of the image we’re trying to start.
* **wait:**This has a boolean value of either true or false. If true, it waits for the instance to reach the desired state before returning
* **region:**pass the variable that defines the region in which an EC2 instance needs to be created.
* **count:**This parameter specifies the number of instances that need to be created. In this case, I’ve only mentioned only one but this depends on your requirements.
* **vpc\_subnet\_id:** pass the subnet id in which you wish to create the instance
* **assign\_public\_ip:**This parameter has a boolean value. If true like in our case, a public IP will be assigned to the instance when provisioned within VPC.

### ****Q49. Can you copy files recursively onto a target host? If yes, how?****

Yes, you can copy files recursively onto a target host using the copy module. It has a recursive parameter which copies files from a directory. There is another module called synchronize which is specifically made for this.

|  |  |
| --- | --- |
| 1  2  3  4 | - synchronize:      src: /first/absolute/path      dest: /second/absolute/path      delegate\_to: "{{ inventory\_hostname }}" |

### ****Q50. Write a playbook to create a backup of a file in the remote servers before copy.****

This is pretty simple. You can use the below playbook:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | - hosts: blocks  tasks:  - name: ansible copy file backup example  copy:  src: ~/helloworld.txt  dest: /tmp  backup: yes |

--------------------------------------------------------------------------------------------------------------------

**Q1) What is Ansible?**

Ansible is developed in [Python language](https://mindmajix.com/python-tutorial). It is a software tool. It is useful while deploying any application using ssh without any downtime. Using this tool one can manage and configure software applications very easily.

**Q2) Ansible Playbooks vs Roles**

|  |  |
| --- | --- |
| **Roles** | **Playbooks** |
| Roles are reusable subsets of a play. | Playbooks contain Plays. |
| A set of tasks for accomplishing a certain role. | Mapps among hosts and roles. |
| Example: common, webservers. | Example: site.yml, fooservers.yml, webservers.yml. |

**Q3) What are the advantages of using Ansible?**

The main three advantages of using this tool are,i.e. Ansible

1. Agentless
2. Very low overhead
3. Good performance

### Q4) Compare Ansible VS Puppet

|  |  |
| --- | --- |
| **Ansible** | **Puppet** |
| Simplest Technology | Complex Technology |
| Written in YAML language | Written in Ruby language |
| Automated workflow for Continuous Delivery | Visualization and reporting |
| Agent-less install and deploy | Easy install |
| No support for Windows | Support for all major OS’s |
| GUI -work under progress | Good GUI |
| CLI accepts commands in almost any language | Must learn the Puppet DSL |

### Q5) How does Ansible Works?

There are many similar automation tools available like [Puppet](https://mindmajix.com/puppet-tutorial#puppet), Capistrano, Chef, Salt, Space Walk, etc, but Ansible categorizes into two types of servers: controlling machines and nodes.

The controlling machine, where Ansible is installed and Nodes are managed by this controlling machine over SSH. The location of nodes is specified by the controlling machine through its inventory.

The controlling machine (Ansible) deploys modules to nodes using SSH protocol and these modules are stored temporarily on remote nodes and communicate with the Ansible machine through a JSON connection over the standard output.

Ansible is agent-less, which means no need for any agent installation on remote nodes, so it means there are no background daemons or programs are executing for Ansible when it’s not managing any nodes.

Ansible can handle 100’s of nodes from a single system over an SSH connection and the entire operation can be handled and executed by one single command ‘ansible’. But, in some cases, where you required to execute multiple commands for a deployment, here we can build playbooks.

Playbooks are a bunch of commands which can perform multiple tasks and each playbook are in YAML file format.

|  |
| --- |
|  |

### Q6) What’s the Use of Ansible?

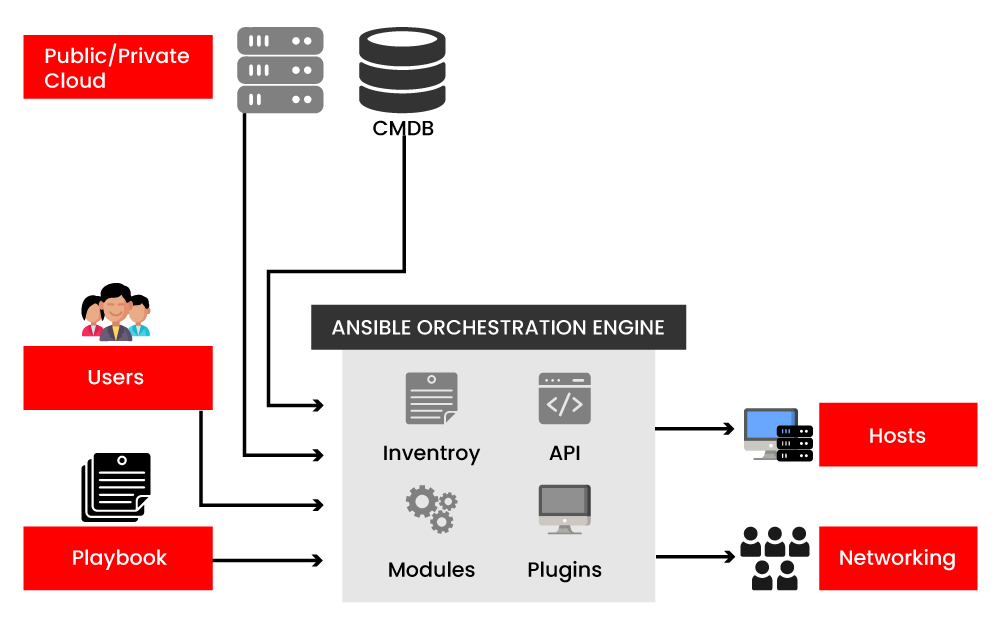
Ansible can be used in IT Infrastructure to manage and deploy software applications to remote nodes. For example, let’s say you need to deploy a single software or multiple software to 100’s of nodes by a single command, here ansible comes into the picture, with the help of Ansible you can deploy as many applications to many nodes with one single command, but you must have a little programming knowledge for understanding the ansible scripts.

We’ve compiled a series on Ansible, title ‘Preparation for the Deployment of your IT Infrastructure with Ansible IT Automation Tool‘, through parts 1-4 and covers the following topics.

### Q7) Explain Ansible architecture?

Ansible automation engine is the main component of Ansible, which interacts directly with the configuration management database, cloud services, and various users who write playbooks to execute it.

The below figure depicts the Ansible architecture:



The following are the components of the Ansible Automation engine:

* **Modules:** Ansible works effectively by connecting nodes and pushing out scripts called "Ansible modules". It helps to manage packages, system resources, files, libraries, etc.
* **Inventories:** These are the lists of nodes or hosts containing their databases, servers, IP addresses, etc.
* **APIs:** These used for commuting public or private cloud services.
* **Plugins:** Plugins augment Ansible's core functionality. Also offers extensions and options for the core features of Ansible - transforming data, connecting to inventory, logging output, and more.
* **Playbooks:** Describes the tasks that need to be executed. They are simple code files written in YAML format and can be used to declare configurations, automating tasks, etc.
* **Hosts:** Hosts are node systems that are automated by Ansible on any machine like Linux, RedHat, Windows, etc.
* **Networking:** Ansible can be used to automate multiple networks and services. It uses a secure and simple automation framework for IT operations and development.
* **Cloud:** A system of remote servers that allows you to store, manage, and process data, rather than a local server.
* **CMDB:** It is a type of repository which acts as a data warehouse for IT installations.

### Q8) What is CI/CD? And how Ansible is related to it?

CI/CD is one of the best software development practices to implement and develop code effectively. CI stands for Continuous Integration, and CD stands for continuous delivery. Continuous Integration is a collection of practices that drive developers to implement and check-in code to version control repositories. Continuous delivery picks up where continuous Integration ends. This process builds software in such a way that software will be released into production at any given time.

Ansible is an excellent tool for CI/CD processes, which provide a stable infrastructure to a provision target environment and then deploy the application to it.

### Q9) Can you create reusable content with Ansible?

Yes, Ansible has the concept of roles that helps to create reusable content. To create a role, you need to follow Ansible's conventions of structuring directories and naming files.

### Q10) Is Ansible a Configuration management tool?

Configuration management is the practice to handle updates and manage the consistency of a product's performance over a particular period of time. Ansible is an open-source IT Configuration Management tool, which automates a wide variety of challenges in complex multi-tier IT application environments.

### Q11) What are the differences between the variable name and environment variables?

|  |  |
| --- | --- |
| **Variable Names** | **Environment Variables** |
| By adding strings, we can build variable names | By accessing existing variables, we can access environment variables |
| Supports adding more strings | The advanced playbooks section sets the environment variables. |
| Use the IPV4 address for variable names. | Use {{ansible\_env.SOME\_VARIABLE}} for remote environment variables |

### Q12) How to create an empty file with Ansible?

To create an empty file, Ansible uses a file module. For this, we need to set up two parameters.

1. **Path**- This place represents the location where the file gets created, either the relative or an absolute path. Also, the name of the file includes here.
2. **State**- For creating a new file, this parameter should be set to touch.

### Q13) How will you set the environment variable or any path for a task or entire playbook?

To set the environment variables, we use the environment keyword. We'll use it at the task or other levels in the play:

environment:

PATH: "{{ ansible\_env.PATH }}:/thingy/bin"

SOME: value

### Q14) How would you describe yourself in terms of what you do and how you’d like to be remembered?

Obviously, I’d like to be remembered as a master of prose who forever changed the face of literature as we know it, but I’m going to have to settle for being remembered as a science fiction writer (and, more and more, critic) who wrote the occasional funny line and picked up a few awards.

### Q15) Why are you attracted to science and science fiction?

Early imprinting, maybe, for the science fiction. When I was quite small a family friend let me read his 1950s run of ‘Galaxy’ magazine. My favorite aunt pressed John Wyndham’s ‘The Day of the Triffids’ on me; a more terrifying great-aunt gave me G.K. Chesterton’s fantastic novels; and so on.

The incurable addiction had begun. Meanwhile, science classes just seemed to be the part of a school that made the most sense, and I fell in love with Pelican pop-maths titles – especially Kasner’s and Newman’s ‘Mathematics and the Imagination’ and all those books of Martin Gardner’s ‘Scientific American’ columns.

### Q16) Tell us about your software company and what sort of software it produced(s).

This goes back to the 1980s and the Apricot home computers, the early, pretty, and non-PC-compatible ones. My pal Chris Priest and I both used them for word processing, and he persuaded me to put together a disk of utilities to improve the bundled ‘SuperWriter’ w/p, mostly written in Borland Turbo Pascal 3 and later 4: two-column printing, automated book index preparation, cleaning the crap out of the spellcheck dictionary, patching SuperWriter to produce dates in UK format, and so on.

Then I redid the indexing software (‘AnsibleIndex’) in CP/M for the Amstrad PCW and its Locoscript word processors. When the Apricot market collapsed, I wrote an Apricot emulator in assembler so that people could keep using their horrible but familiar old software on a PC. Eventually, in a fit of nostalgia, I collected all my columns for ‘Apricot File’ and various Amstrad PCW magazines as books unoriginally titled ‘The Apricot Files’ and ‘The Limbo Files’. (That’s probably enough self-promotion, but there’s lots more at <https://ansible.uk/>.)

### Q17) Describe your newsletter Ansible and who it’s aimed at.

It appears monthly and has been called the ‘Private Eye’ of science fiction, but isn’t as cruel and doesn’t (I hope) recycle old jokes quite as relentlessly. Though I feel a certain duty to list some bread-and-butter material like conventions, award winners, and deaths in the field, ‘Ansible’ skips the most boring SF news – the long lists of books acquired, books published, book sales figures, major new remainders – in favor of quirkier items and poking fun at SF notables. The most popular departments quote terrible lines from published SF/fantasy and bizarre things said about SF by outsiders (‘As Others See Us’). All the back issues of ‘Ansible’ since it started in 1979 can be read online.

### Q18) So how does Ansible work? Please explain in detail?

Within the market, they are many automation tools like Puppet, Capistrano, Chef, Salt, Space Walk, etc.

* When it comes to Ansible, this tool is categorized into two types of servers:  
      1. Controlling machines  
      2. Nodes
* Ansible is an agentless tool so it doesn’t require any mandatory installations on remote nodes. So there are no background programs that are executed while it is managing any nodes.
* Ansible is able to handle a lot of nodes from a single system over an SSH connection.
* Playbooks are defined as a bunch of commands where they are capable of performing multiple tasks and they are in YAML file format.

### Q19) What does Ansible offer?

 Ansible  offers:

* Security and Compliance policy integration
* Automated workflow for Continuous Delivery
* Simplified orchestration
* App deployment
* Configuration management
* Streamlined provisioning

|  |
| --- |
|  |

### Q20) Can we manage Windows Nano Server using Ansible?

No, it is not possible to manage Windows Nano Server using Ansible as it doesn't have full access to the .Net framework, which is primarily used by internal components and modules.

### Q21) Do we have any Web Interface/ Rest API etc fo

Yes, Ansible Inc makes a great efficient tool. It is easy to use.

### Q22) What is Ansible Tower?

Ansible is classified as a web-based solution which makes Ansible very easy to use. It is considered to be or acts like a hub for all of your automation tasks. The tower is free for usage till 10 nodes.

### Q23) What are the features of the Ansible Tower?

Features of the Ansible Tower are:

* Ansible Dashboard
* Real-time job status updates
* Multi-playbook workflows
* Who Ran What Job When
* Scale capacity with tower clusters
* Integrated notifications
* Schedule ansible jobs
* Manage and track inventory
* Remote command execution
* REST API & Tower CLI Tool

### Q24) How do change the documentation and submit it?

Usually, the documentation is kept in the main project folder in the git repository. For complete instructions on this can be available in docs.

## Ansible Interview Questions for Experienced

### Q25) How do you access Shell Environment Variables?

If you are just looking to access the existing variables then you can use the “env” lookup plugin.

For example:

Accessing the value of Home environment variable on management machine:

local\_home:”{{lookup(‘env’,’HOME’)}}”

### Q26) How can you speed up management inside EC2?

It is not advised to manage a group of EC2 machines from your laptop. The best way is to connect to a management node inside Ec2 first and then execute Ansible from there.

### Q27) Is it possible to increase the Ansible reboot module to more than 600 seconds?

Yes, it is possible to increase the Ansible reboot module to specific values using the below syntax:

- name: Reboot a Linux system

reboot:

reboot\_timeout: 1000

### Q28) How can you use docker modules in Ansible?

Docker modules require docker-py installed on the host running Ansible.

$ pip install 'docker-py>=1.7.0'

The docker\_service module also requires docker-compose

$ pip install 'docker-compose>=1.7.0'

### Q29) Explain how you will copy files recursively onto a target host?

The copy file in Ansible has a recursive parameter. If you have to copy files for a large number of files, then the synchronizing module is the best choice for it.

- synchronize:

src: /first/absolute/path

dest: /second/absolute/path

delegate\_to: "{{ inventory\_hostname }}"

### Q30) How can you disable cowsay?

If cowsay is installed then executing your playbooks within Ansible is very smooth.

Even if you think that you want to work in a professional cow free environment, then you will have two options:

1. Uninstall cowsay
2. Setting up value for the environment variable, like below

Export ANSIBLE\_NOCOWS=1

### Q31) How can you access a list of Ansible\_Variables?

By default, Ansible gathers facts under machines under management. Further, these facts are accessed in Playbooks and in templates. One of the best ways to view a list of all the facts that are available in a machine, then need to run the setup module in the ad-hoc way:

Ansible- m setup hostname

Once this statement is executed, it will print out a dictionary of all the facts that are available for that particular host. This is the best way to access the list of Ansible\_variables.

### Q32) How can you see all the variables specific to my host?

To see all the host-specific variables, that include all facts and other resources are:

Ansible - m debug- a “var=hostvars[‘hostname’]” localhost

### Q33) How do you access a variable name programmatically?

By adding strings together, the variables names are built programmatically like below format:

{{ hostvars[inventory\_hostname]['ansible\_' + which\_interface]['ipv4']['address'] }}

'inventory\_hostname' is a variable that represents the present host you are looping over.

### Q34) How to configure a jump host for accessing servers that have no direct access?

We should set a ProxyCommand in the ansible\_ssh\_common\_args inventory variable. For connecting to the relevant host, arguments defined in this variable are added to scp/ssh/sftp command line.

For example,

[gatewayed]

foo ansible\_host=192.0.2.1

bar ansible\_host=192.0.2.2

With the following contents, create the group\_vars/gatewayed.yml

ansible\_ssh\_common\_args: '-o ProxyCommand="ssh -W %h:%p -q user@gateway.example.com"'

When connecting to any hosts in the group gatewayed, Ansible will append these arguments to the command line.

### Q35) Explain how you can generate encrypted passwords for the user module?

Ansible ad-hoc command is the easiest option:

ansible all -i localhost, -m debug -a "msg={{ 'mypassword' | password\_hash('sha512', 'mysecret') }}"

The mkpasswd utility available on the Linux systems is also the best option:

mkpasswd --method=sha-512

### Q36) Can you keep data secret in the playbook?

Yes. If any task that you want to keep secret in the playbook when using -v (verbose) mode, the following playbook attribute will be helpful:

- name: secret task

shell: /usr/bin/do\_something --value={{ secret\_value }}

no\_log: True

It hides sensitive information from others and provides the verbose output.

### Q37) What is idempotency?

Idempotence is an essential feature of Ansible, which helps you to execute one or more tasks on a server as many times as needed, but without changing the result beyond the initial application.

### Q38) Can you create encrypted files with Ansible?

Yes, using the 'ansible-vault create' command, we can create encrypted files

$ ansible-vault create filename.yaml

### Q39) What is the difference between a playbook and a play?

A playbook is a list of plays. A play is a set of tasks and roles that run on one or more managed hosts. Play includes one or more tasks.

### Q40) How will you get access to the ansible host when I delegate a task?

We can access it through host variables and even works for all the overridden variables like ansible\_port, ansible\_user, etc.

original\_host: "{{ hostvars[inventory\_hostname]['ansible\_host'] }}"

### Q41) Explain the Ansible Tag's usage?

A tag is an attribute that sets the ansible structure(plays, tasks, roles). When there's an extensive playbook needed, it's more useful to run just a part of it as opposed to the entire thing. That's where tags usage is required.

### Q42) How can you filter out tasks in tags?

* Use –tags or –skip-tags options on the command line
* Use the TAGS\_RUN and TAGS\_SKIP options, If you're in Ansible configuration settings.

### Q43) What are handlers?

In Ansible, handlers are just like normal tasks in a playbook but run when tasks include the notify directive and also indicate that it changed something. It runs only once after all the tasks executed in a particular play. It automatically loads through roles/<role\_name>/handlers/main.yaml.

They are used to trigger the status of a service, such as restarting or stopping a service.

### Q44) How will you upgrade Ansible?

Using the command "sudo pip install ansible==<version-number>", we can easily upgrade Ansible.

### Q45) Ansible vs Chef?

|  |  |
| --- | --- |
| **Ansible** | **Chef** |
| Ansible is easier to set up and provides faster performance | Compared to Ansible, Chef is not very easy to set up |
| Ansible uses YAML (Python) for managing configurations | Chef uses DSL (Ruby) for managing configurations |
| Highly scalable | Highly scalable |
| Ansible charges annually $10,000 | Chef Automate charges an annual fee of $13700 |

### Q46) Why don’t you ship in X format?

They are several reasons for not shipping in X format. In general, it caters to maintainability. Within the market, they are tons of different ways to ship software and it is very tedious to support all of them.

### Q47) What is Ansible can do?

Ansible can do the following for us:

1. Configuration management
2. Application deployment
3. Task automation
4. IT orchestration

### Q48) Please define what is Ansible Galaxy?

Ansible Galaxy refers to the website Galaxy where the users will be able to share all the roles to a CLI ( Command Line Interface) where the installation, creation, and managing of roles happen

### Q49) Can you explain how to handle various machines requiring different user accounts or ports to log in?

Just by setting inventories in the inventory file, we can handle various machines requiring different user accounts or ports to log in.

For example, the following hosts have different ports and usernames:

[webservers]

asdf.example.com ansible\_port=5000 ansible\_user=alice

jkl.example.com ansible\_port=5001 ansible\_user=bob

You can specify the connection type to be used by:

[testcluster]

localhost ansible\_connection=local

/path/to/chroot1 ansible\_connection=chroot

foo.example.com ansible\_connection=paramiko

File them in a group\_vars/<group-name> file.

### Q50) Do you know what language Ansible is written in?

Ansible is written in Python and PowerShell

### Q51) Please explain what is Red Hat Ansible?

Ansible and Ansible Tower by Red Hat, both are an end to end complete automation platforms which are capable of providing the following features or functionalities:

* Provisioning
* Deploying applications
* Orchestrating workflows
* Manage IT systems
* Configuration of IT systems
* Networks
* Applications

All of these activities are dealt with by Ansible where it can help the business to solve real-time business problems.

### Q52) Is Ansible is an open-source tool?

Yes, Ansible is an open-source tool that is a powerful automation software tool that one can use.

### Q53) Why you have to learn Ansible?

Ansible is more a tool for servers but does it have anything for networking. If you closely look into it, there is some support available in the market for networking devices. Using this tool, it will give you an overall view of your environment and also knowledge of how it works when it comes to network automation.

It is one of those tools where it is considered to be good to explore a new tool.

### Q54) What are Ansible server requirements?

You need to have a virtual machine with Linux installed, which has Python 2.6 version or higher.

### Q55)  How to install Ansible on CentOS?

**Step 1:** Update your Control Node

yum update

**Step 2:** Install the EPEL Repository

yum install epel-release

**Step 3:** Install Ansible

yum install Ansible

### Q56) How can you connect to other devices within Ansible?

Once, Ansible is installed and the basic setup has been completed, an inventory is created. This would be the base and one can start testing ansible. To connect to a different device then you have to use “Ping module”. This can be used as a simple connection test.

Ansible - m ping all

### Q57) Can you build your own modules with Ansible?

Yes, we can create or own modules within Ansible.

It is an open-source tool that primarily works on Python. If you are good at programming in Python you can start creating your own modules in few hours from scratch and you don't need to have any prior knowledge of the same.

### Q58) How can you find information in Ansible?

After completing the basic setup, one has to make sure to find out the module called the “setup” module. Using this setup module, you will be able to find out a lot of information.

### Q59) What does Fact mean in Ansible?

The term “Facts” is commonly used in an Ansible environment. They are described in the playbook areas where it displays known and discovered variables about the system.  Facts are used to implement conditional executions and also used for getting ad-hoc information of information.

You can see all the facts via:

$ ansible all- m setup

So if you want to extract only a certain part of the information then you can use the “setup” module where you will have an option to filter out the output and just get hold of the fact that you are in need of.

### Q60) What is ask\_pass in ansible?

The ask\_pass is controlled in Ansible Playbook.

This controls whether ansible-playbook prompts a password by default. Usually, the default behavior is no:

It is always set to ask\_pass=True

If you are using SSH keys for authentication purposes then you really don’t have to change this setting at all.

### Q61) Explain What is ask\_sudo\_pass

This control is very similar to ask\_pass

The ask\_sudo\_pass controls the Ansible Playbook to prompt a Sudo password. Usually, the default behavior is no:

ask\_sudo\_pass= True

One has to make sure and change this setting where the sudo passwords are enabled most of the time.

### Q62) Explain what is ask\_vault\_pass?

Using this control we can determine whether Ansible Playbook should prompt a password for the vault password by default. As usual, the default behavior is no

ask\_vault\_pass= True

### Q63) Explain Callback\_plugin in Ansible?

Callbacks are explained as a piece of code in ansible environments where to get is used to call a specific event and permit the notifications.

This is more sort of a developer-related feature and allows low-level extensions around ansible so that they can be loaded from different locations without any problem.

### Q64) Explain Module utilities in Ansible?

Ansible provides a wide variety of module utilities that help the developers while developing their own modules. The basic.py is a module that provides the main entry point for accessing the Ansible library and using those as basics one can start off working.

### Q65) Where is the unit testing is available in Ansible?

Unit tests for all the modules are available in .test/units/modules. Firstly you have to set up your testing environment

### Q66) Explain in detail ad-hoc commands?

Well, ad-hoc commands are nothing but a command which is used to do something quickly and it is more sort of one-time use. Unlike, the playbook is used for a repeated action which is something that is very useful in the Ansible environment. But there might be scenarios where we want to use ad-hoc commands which can simply do the required activity and it is a nonrepetitive activity.

**1) What Is Ansible?**

Ansible is a configuration management system. It is used to set up and manage infrastructure and applications. It allows users to deploy and update applications using SSH, without needing to install an agent on a remote system.

**2) What’s the use of Ansible?**

[Ansible](https://www.guru99.com/ansible-tutorial.html) is used for managing IT infrastructure and deploy software apps to remote nodes.

For example, Ansible allows you to deploy as an application to many nodes with one single command. However, for that, there is a need for some programming knowledge to understand the ansible scripts.

**3) What is Ansible Galaxy?**

Ansible can communicate with configured clients from the command line by using ansible command. It also allows you to automate configuration by using ansible-playbook command. To create the base directory structure, you can use a tool bundled with Ansible which is known as ansible-galaxy.

**Command:**

$ ansible-galaxy init azavea. packer

azavea.packer was created successfully

**4) What is Continuous Delivery?**

Continuous delivery is a practice of delivering the software as soon as it developed. In this method, we need to use versioning control system. The software is constantly updated in live production systems.

**5) What is the way to access shell environment variables in Ansible?**

In Ansible, if you want to access existing variables, the user needs to use the ‘env’ lookup plugin. Example, to access the value of the Office environment on the management machine:

You need to write following code:

---

# ...

vars:

local\_home: "{{ lookup('env','Office') }}"

I

{{ ansible\_env.SOME\_VARIABLE }}



**6) What is the code you need to write for accessing a variable name?**

Variable names can be built by adding using the following method:

{{ hostvars[inventory\_hostname]['ansible\_' + which\_interface]['ipv4']['address'] }}

The method of using hostvars is important because it’s a dictionary of the entire namespace of variables. ‘inventory\_hostname’ variable specifies the current host you are looking over in the host loop.

**7) Explain how you can disable cowsay?**

If cowsay is installed then executing playbooks inside the Ansible you can disable coway by using following options:

1. Uninstall cowsay
2. Setting up value for the environment variable

export ANSIBLE\_NOCOWS=1

**8) Explain how you can copy file recursively onto a target host?**

The “copy” module has a recursive parameter. However, if you want this to perform more efficient for a large number of files, then “synchronize” module is the best option for you.

**9) How Can you submit a change to the Documentation in Ansible?**

Documentation for Ansible is kept in the project git repository. It contains complete instructions for contributing can be found in the docs.

**10) What Is the Best Method to Make Content Reusable/redistributable?**

You can read everything about “Roles” in the playbooks documentation section. This helps to make playbook content self-contained and shareable with other ansible users.

**11) What is Ansible Tower?**

Ansible tower is a tool which makes Ansible very easy to use. It acts as a hub for the task automation. The tower is free for usage till 10 nodes.

**12) What’s the method to check the inventory vars defined for the host?**

**For that use this command:**

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

**13) State the difference between Variable name and Environment Variables.**

| **Variable Name** | **Environment Variables** |
| --- | --- |
| It can be built by adding strings. | To access the environment variable, you need to access existing variables. |
| {{ hostvars[inventory\_hostname][‘ansible\_’ + which\_interface][‘ipv4’][‘address’] }} | # … vars: local\_home: “{{ lookup(‘env’,’HOME’) }}” |
| Allows to add strings | To set environment variables, we need to see the advanced playbooks section. |
| Ipv4 address type use for Variable names we use the ipv4 address. | For Remote environment variables, use {{ ansible\_env.SOME\_VARIABLE }} |

**14) What are ad-hoc commands?**

You can think of ad-hoc commands as a way for us to take actions on our hosts without writing a playbook. For example, if we want to reboot all hosts in a particular group(webservers). Then you can write a playbook or simply run a one-off ad-hoc command.

**15) Explain Ansible facts**

You can think of ansible facts as a way for ansible to get information about a host and store them in variables for easy access. This information stored in predefined variables are available to use in the playbook. To generate facts, ansible runs the setup module.

**16) How do you see all variables for a host?**

You can see them using the hostvars variable. This stores host variables with the hostname as key. For example, to look at the variables defined for localhost, you can run;

ansible -m debug -a "var=hostvars[inventory\_hostname]"

**17) Explain modules in ansible**

Modules in Ansible are idempotent. From a RESTful service standpoint, for the operation to be idempotent, clients can perform the same result by using modules in Ansible. Multiple identical requests become a single request.

There are two different types of modules in Ansible:

* **Core modules**
* **Extras modules**

**Core Modules**

The Ansible team maintains these types of modules, and they will always ship with Ansible software. They will also give higher priority for all requests than those in the “extras” repos.

**Extras Modules:**

These modules currently is bundled with Ansible but might available separately in the future. They are also mostly maintained by the Ansible community. These modules are still usable, but it can receive a lower rate of response to issues and pull requests.

**18) When should you test playbooks and roles?**

In ansible, Tests can be added either in new Playbooks or to existing Playbooks. Therefore, most of the testing job offers a clean hosting each time. By using this testing methodology, you need to make very little to no code changes.

1) What is Ansible?

Ansible is an open-source IT engine that automates application deployment, cloud provisioning, intra service orchestration, and other IT tools.

Ansible is very easy to deploy because it does not use any **agents** or **custom security** infrastructure on the client-side, and by pushing modules to the clients. These modules are executed locally on the client-side, and the output is pushed back to the Ansible server. It can easily connect to clients using **SSH-Keys**.

2) What are the uses of Ansible?

* Ansible is free to use by everyone.
* Ansible is very consistent and lightweight, and no constraints regarding the operating system or underlying hardware are present.
* It is very secure due to its agentless capabilities and open **SSH** security features.
* Ansible does not need any special system administrator skills to install and use it.
* Ansible has a smooth learning curve determined by the comprehensive documentation and easy to learn structure and configuration.
* Its modularity regarding **plugins, inventories, modules,** and **playbooks** make Ansible perfect companion orchestrate large environments.

3) What is Ansible Tower?

Ansible Tower is like Ansible at a more enterprise level. It is the web-based solution for managing your organization with an easy user interface that provides a dashboard with all of the state summaries of all the hosts. And allows quick deployments, and monitors all configurations.

### 4) What is Ansible Galaxy?

Ansible Galaxy is a galaxy website where users can share roles and to a command-line tool for **installing, creating,** and **managing** roles.

Ansible Galaxy gives greater visibility to one of Ansible's most exciting features, such as application installation or reusable roles for server configuration. Lots of people share roles in the Ansible Galaxy.

### 5) What is Ansible Modules?

Ansible modules are discrete units of code which can be used from the command line or in a playbook task.

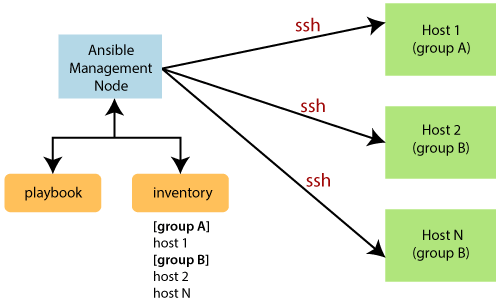
The modules also referred to as task plugins or library plugins in the Ansible.

Ansible ships with several modules that are called **module library**, which can be executed directly or remote hosts through the playbook.

Users can also write their modules. These modules can control like **services, system resources, files,** or **packages,** etc. and handle executing system commands.

### 6) How does Ansible work?

Ansible works by connecting to the nodes and pushing out a small program called **Ansible modules** to them. Then Ansible executed these modules and removed them after finished. The library of the modules can reside on any machine, and there are no **daemons, servers,** or **databases** required.



The **Management Node** is the controlling node that controls the entire execution of the playbook. The **inventory** files provide the list of hosts where the Ansible modules need to be run. The **Management Node** makes an **SSH** connection and executes the small modules on the **host's** machine and install the software.

7) What are the Ansible Server requirements?

* If you are a windows user, then you required to have a virtual machine in which Linux should be installed.
* And it requires python 2.6 version or higher.

8) What are the variables in Ansible?

The variable is very similar to using the variables in a programming language. It helps you to assign a value to a variable and use it anywhere in the playbook. You can put the conditions around the value of the variables and use them in the playbook accordingly.

9) What is Ansible Task?

Ansible Task allows us to break up bits of configuration policy into smaller files. These are the block of code which can be used to automate any process.

10) Explain the basic terminologies or concepts in Ansible?

Some basic terms which are commonly used in Ansible, such as:

* **Controller Machine:** The Controller machine is used to provisioning the servers, which is managed. This is the machine where Ansible is installed.
* **Inventory:** An inventory is an initialization file which has details about the different servers you are managing.
* **Playbook:** It is a code file that is written in the YAML format. A playbook contains the tasks that need to be automated or executed.
* **Task:** Every task represents a single procedure that needs to be executed, e.g., installs a library.
* **Module:** A module is the set of tasks that can be executed. Ansible has 100s of built-in modules, and also you can create custom ones.
* **Role:** The role is a pre-defined way for organizing playbooks and other files to facilitate sharing and reusing portions of provisioning.
* **Play:** The task executed from start to finish, or the execution of a playbook is called the play.
* **Facts:** Facts are global variables which are store details about the system, such as network interfaces or operating system.
* **Handlers:** Handlers are used to trigger the status of a service, such as restarting or stopping a service.

11) What are Ad-hoc commands?

Ad-hoc commands are simple one-line commands which are used to perform a certain task. You can think of Ad-hoc commands as an alternative to writing playbooks. An example of an Ad-hoc command, as shown as follows:

1. ansible host -m netscaler -a "nsc\_host=nsc.example.com user=ansiuser password=ansipass"

2) How would you access a variable of the first host in a group?

Executes the following commands to access a variable of the first host in a group:

1. {{ hostvars[groups['webservers'][0]]['ansible\_eth0']['ipv4']['address'] }}

This command is accessing the hostname of the first machine in the webservers group. If you are doing this by a template, then use the jinja2 '#set', or you can use set\_fact.

13) How to keep secrete data in a playbook?

The following command is used to keep verbose output but hide the sensitive information from others who would like to be able to see the output.

1. -   name: secrete task
2. shell: /user/bin/do\_somthing -value={{secrete value}}
3. no\_log: True

Also, the no\_log attribute can apply to an entire play.

1. -   hosts: all
2. no\_log: True

14) What are Ansible Vaults, and why are they used?

Ansible Vault is a feature that allows keeping all your secrets safe. It can encrypt entire files, YAML playbooks, or even a few variables. It provides a facility where you also can integrate the sensitive data into your playbooks.

Vault is implemented with file-level granularity, where the files are completely encrypted or unencrypted. It uses the same password for encrypting as well as decrypt the data, which makes Ansible vault user-friendly.

15) What is the way to access shell environment variables in Ansible?

In Ansible, if you want to access existing variables, then you need to use the '**env**' lookup plugin.

**For example,** you want to access the value of the Office environment on the management machine, as shown in the following command, such as:

1. ---
2. # ...
3. vars:
4. local\_home: "{{ lookup('env','Office') }}"
5. I
6. {{ ansible\_env.SOME\_VARIABLE }}

16) Who you can copy file recursively onto a target host?

The "copy" module has a recursive parameter. Therefore, if you want to perform more efficient for a large number of files, then the "synchronize" module is the best option for doing this task.

17) How do you set the path or any other environment variable for a task?

The environment variables can be set by using the 'environment' keyword. It can be set for either a task or an entire playbook. Follow the following code to see how do set the path, such as:

1. environment:
2. PATH: "{{ansible\_env.PATH}}:/thingy/bin"
3. SOME: value

18) How can you see all variables for a host?

You can see all the variables using the **host vary** variable. It stores host variables with the hostname as key. For example, to look at the variables defined for localhost, you can run the following command, such as:

1. ansible -m debug -a "var=hostvars[inventory\_hostname]"

19) What is the difference between the Variable Name and Environment Variable in Ansible?

|  |  |
| --- | --- |
| **Variable Name** | **Environment Variable** |
| You need to add strings to create variable names. | You need existing variables to access environment variables. |
| You can easily create multiple variable names by adding strings. | To create environment variables, you must refer to the advanced Ansible playbook. |
| Use the IPV4 address for variable names. | Use {{ansible\_env.SOME\_VARIABLE}} for remote environment variables. |

20) What is RedHat Ansible?

Ansible and Ansible Tower by Red Hat, both are the end to end complete automation platforms which are capable of providing the following features or functionalities:

## Q1. What is Ansible?

**Ans.** Ansible is an open-source automation tool used for application deployment, configuration management, and cloud provisioning. It automates the entire IT environment to simplify complex tasks and make developers’ jobs easy and manageable.

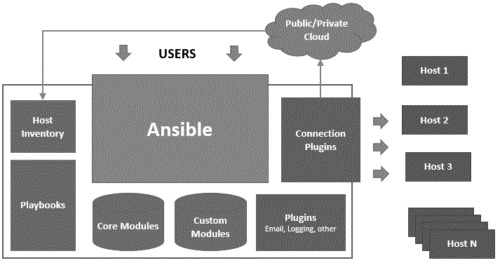
It also works on IT orchestration, where we can run tasks in sequence and create a chain of events that appears on different servers or devices. Ansible is growing faster than other automation tools. Here are some of the advantages of using Ansible:

* It is a free and open-source tool
* Ansible is a simple tool to set up and use
* It helps to model highly complex workflows
* We can customize the whole application according to the need

***To learn about DevOps, read our blog –***[***What is DevOps***](https://www.naukri.com/learning/what-is-devops-development-and-operations-st607-tg387)***?***

## Q2. How does Ansible work?

**Ans.** Ansible does not break its process into steps or divisions. To understand how Ansible works, we have to follow stages:



### ****Setting up the architecture:****

Ansible requires nodes and release small programs called “Ansible Modules.” These released modules act as a resource to the desired state of the system. It executes these modules and removes them after the execution.

**SSH Keys:**There is an inventory file in the controlling machine that occupies the location of the node system. It helps modules to locate on the node system by running the playbook on the controlling system.

Example:

**ssh-agent bash**

**ssh-add ~/.ssh/id\_rsa**

**Managing Inventory:** Ansible keeps all the machines that it is operating on the system in the INI file. The machines in the INI file are grouped as per the user’s choice. New machines are also added without any SSL signing. It is also plugging to sources such as Openstack, EC2, Rackspace, and others. Here, how inventory looks like:

**[webservers]**

**www01.example.com**

**www02.example.com**

**[dbservers]**

**db01.example.com**

**Db02.example.com**

**Using Ansible:**We can use Ansible, once the instance is available. All the requirements in the Ansible running commands and resource module are taken care of. Ansible already consists of the arsenal module.

Example:

**ansible all -m ping**

**ansible foo.ex.com -m yum -a “name=httpd state=installed”**

**ansible foo.ex.com -a “/usr/sbin/reboot1”**

*Explore popular courses on Naukri Learning:*

|  |  |
| --- | --- |
| [Popular Python Courses](https://www.naukri.com/learning/python-courses-certification-training-st619-tg21) | [Popular DevOps Courses](https://www.naukri.com/learning/devops-courses-certification-training-st607-tg387) |
| [Top JavaScript Courses](https://www.naukri.com/learning/javascript-courses-certification-training-st619-tg305) | [Top Java Courses](https://www.naukri.com/learning/java-courses-certification-training-st619-tg17) |

## Q3. What are the requirements of Ansible server?

**Ans.**If you are a windows user then it is required to have a virtual machine in which Linux can be installed.

## Q4. Explain different components of Ansible?

**Ans.** Following are some components:

* Inventory
* Playbooks
* Plays
* Tasks
* Modules
* Roles
* Handlers
* Facts
* Templates
* Variables

## Q5. Is Ansible an open-source tool?

**Ans.** Yes, Ansible is an open-source tool because we can rewrite the modules. An open-source automated engine provides a source to automate apps as per the need.

## Q6. Name the language in which Ansible is written?

**Ans.** Ansible is written in Python and Powershell.

***Want to learn about Python, read our blog –***[***What is Python***](https://www.naukri.com/learning/what-is-python-st619-tg21)***?***

## Q7. Differentiate between Ansible Playbooks and Roles.

**Ans.**Following are the differences between Roles and Playbooks:

|  |  |
| --- | --- |
| **Roles** | **Playbooks** |
| It is a set of tasks and additional files to configure the host | It is a mapping between roles and hosts |
| Roles are considered as group tasks into one container. We  can use role for setting up MySQL, and for setting up postfix | A playbook identifies what is happening where. Sometimes playbook only contains single play, but we can get the number as required |
| Example: Common, Webservers, etc. | Example: fooservers.yml, site.yml, webservers.yml, etc. |

## Q8. What are the variables in Ansible?

**Ans.** Variables are alike with the variables in another programming language. These are assigned a value that is used to determine playbooks. We can also use conditions by using variables:

**– hosts: your host01**

**vars:**

**port\_Tomcat: 2050**

Here, a variable port Tomcat is defined, and the value assigned to the port number is 2020.

**Also Read:**[**Most In Demand Tech Skills to Master**](https://www.naukri.com/learning/articles/trending-tech-skills-to-master/)

## Q9. What are the differences between a variable name and an environment variable?

**Ans.**Following are the difference between a Variable Name and an Environment Variable:

|  |  |
| --- | --- |
| **Variable Name** | **Environment Variable** |
| * Strings need to be added to create variable names * We can define multiple variable names by adding strings * Ipv4 address is used for variable names | * An existing variable is needed to access the environment variable * We have to use the Ansible playbook to create an environment variable * Use **{{ansible\_env.SOME\_VARIABLE\_A }}** for remote environment variables |

## Q10. What are the Ansible Modules? Explain the different types.

**Ans.** This is a commonly-asked Ansible interview question. Modules are the small units of code that perform a specific task. It can be used from the command line or in a playbook task. It helps to automate a wide range of tasks.

**ansible webservers -m service -a “name=httpd state=installed”**

**ansible webservers -m ping**

**ansible webservers -m command -a “/sbin01/reboot -t now”**

There are two types of modules:

* **Core modules:**The core team maintains these modules, and they will ship with Ansible only. They receive slightly high priority for the requests than extras modules.
* **Extras modules:** Ansible itself ships these modules, but they can be shipped separately in the future. Non-core modules receive a lower response as compare to core modules.

## Q11. How can you access a variable of the first host in a group?

**Ans.** By using the following command, we can access a variable:

{{ hostvars[groups[‘webservers’][0]][‘ansible\_eth0’][‘ipv4’][‘address’] }}

In the given code, we’re accessing the hostname of the first machine in the webservers group. To use a template to do this, we can use Jinja2 ‘#set’ or set\_fact, like written below:

**– set\_fact: headnode={{ groups[[‘webservers’][0]] }}**

**– debug: msg={{ hostvars[headnode].ansible\_eth0.ipv6.address }}**

## Q12. How to write an Ansible Handler with multiple tasks?

**Ans.** If you want to create a handler that restarts a service when it is running

Ansible Handlers. It can listen to the general topics, and those topics are shown below. This task makes it easier to trigger multiple handlers. It also decouples handler from their name and makes it simpler to share handlers in Playbooks and roles:

**– name: Check if restarted**

**shell: check\_is\_var.sh**

**register: result**

**listen: Restart processes\_a**

**– name: Restart conditionally step 1**

**service: name=service state=restarted**

**when: result**

**listen: Restart processes\_a**

**Explore the**[**Top Online IT Courses**](https://www.naukri.com/learning/articles/top-online-courses-for-it-professionals/)

## Q13. How to keep secret data in the playbook?

**Ans.** If we have a task and we don’t want to show the output or command given in it when using -v (verbose) mode, the following code is used:

**– name: secret task**

**shell: /usr/bin\_a/do\_something –value={{ secret\_value01 }}**

**no\_log: True**

It can be used to keep verbose output

**– hosts: all**

**no\_log: True**

## Q14. What is Ansible Tower?

**Ans.** Ansible Tower is a web-based solution used by IT teams for managing an organization with a very easy user interface. It is mainly designed for solving all sorts of automation tasks. Ansible Tower provides a dashboard with the state summaries of all the hosts, monitors each configuration, and allows fast deployments.

It allows to share the SSH credentials, manage inventories graphically, logs all the jobs, and syncs them with a variety of cloud providers.

**Explore**[**Popular Online Courses**](https://www.naukri.com/learning/)

## Q15. Mention the features of Ansible Tower.

**Ans.** Following are the features:

* **Ansible Tower Dashboard –** It presents all sorts of things available on their environments such as the hosts, the recent job activity, inventory status, and more.
* **Real-Time Job Updates –** An Ansible can automate the whole infrastructure so that we can see the real-time job updates, like tasks and plays divided by each machine. By this, we can check our automation status and easily track what’s next in the queue.
* **Multi-Playbook Workflows –** It allows to connect any number of playbooks, in spite of using different inventories, run various users, or utilize multiple credentials.
* **Scale Capacity With Clusters –** With this feature of the Ansible tower, we can connect various nodes of it into the Ansible tower cluster. It creates redundancy that allows scaling Ansible automation over the enterprise.
* **Integrated Notifications –** This feature notifies if the job succeeds or fails in the whole organization or customize on a per-job basis.
* **Remote Command Execution –** On this platform, we can run simple tasks like restart malfunctioning service, reset passwords, and add users in the inventory with Ansible Tower’s remote command execution.
* **Schedule Ansible Jobs –**Ansible Tower provides this feature for various kinds of jobs like Playbook runs, source control, and cloud inventory updates that can be scheduled to run across the platform according to the need.
* **Who Ran What Job When** – It provides us the option to know who ran what job when and whereas all the automation activity is securely logged in the Ansible tower.
* **Manage & Track Inventory –** It provides the benefit of managing the entire infrastructure by allowing us to use inventory from public cloud providers like Microsoft Azure, Amazon web services, and others.
* **Self-Service –** It allows us to launch Playbooks with just a single click. It also helps to choose from the available credentials and record the available deployments.

## Q16. Write a code to set the PATH or any environment variable for a task?

**Ans.**Environment variables can be set by using the keyword “Environment.” This variable can be set for the entire Playbook or only for a particular task.

Following code shows, how to set a path:

**Environment:**

**PATH: “{{ ansible\_env.PATH }}:/thing\_a/bin”**

**SOME: value01**

***Learn about Cloud Computing, read our blog –***[***What is Cloud Computing***](https://www.naukri.com/learning/what-is-cloud-computing-st607-tg185)***?***

## Q17. How can you handle various machines needing different user accounts or ports to log in with?

**Ans.** It can be handled by setting inventory variables into the inventory file.

These hosts consist of different username and ports:

**[webservers]**

**www.example.com ansible\_port=2050 ansible\_user=tom**

**jkl.examplein ansible\_port=5002 ansible\_user=jack**

We can also specify the connection type:

**[testcluster]**

**localhost ansible\_connection=local**

**/path/to\_a/chroot1 ansible\_connection=xyz**

**www.example.com ansible\_connection=abc**

To make the task easier, we can compile it into a file **group\_vars/<group-name>** file.

## Q18. Does Ansible support AWS?

**Ans.**Ansible has multiple modules that support AWS; some of them are:

* Route53
* Elastic Cloud Compute (EC2)
* Virtual Private Cloud (VPC)
* Relational Database Service (RDS)
* CloudWatch
* Identity Access Manager (IAM)
* DynamoDB
* ElastiCache
* CloudTrail
* CloudFormation
* Simple Storage Service (S3)
* Autoscaling groups

**Also Read:**[***What is AWS Certification?***](https://www.naukri.com/learning/what-is-aws-certification-st607-tg105)

## Q19. Can you copy files to remote locations recursively onto a target host? If yes, how?

**Ans.** Yes, we can copy files to remote locations repetitively by using a copy module. It can copy files from the directory recursively, and the other module called synchronize that is specially used for this process.

**– synchronize:**

**src: /first/absolute01/path**

**dest: /second/absolute02/path**

**delegate\_to: “{{ inventory\_hostname\_xyz }}”**

***Check out the***[***best Python Courses online***](https://www.naukri.com/learning/python-courses-certification-training-st619-tg21)

## Q20. How can you access a variable of the first host in a group?

**Ans.**This is one of the important Ansible interview questions. We can access a variable by executing the below command:

{{ hostvars[groups[‘webservers’][0]][‘ansible\_eth0’][‘ipv4’][‘address’] }}

In the given command, we are obtaining the hostname of the first machine in the webservers group. We can use Jinja2 ‘#set’ or set\_fact that is shown below:

**– set\_fact: headnode={{ groups[[‘webservers’][1]] }}**

**– debug: msg={{ hostvars[headnode].ansible\_eth0.ipv6.address }}**

**Check out the popular**[***DevOps Courses***](https://www.naukri.com/learning/devops-courses-certification-training-st607-tg387)

## Q21. What is CD/CI in Ansible?

**Ans.** CD (Continuous Deployment) and CI (Continuous Integration) are approaches in software development that help you deliver high-quality applications. CI/CD enables incremental code changes from the developer to be delivered quickly and reliably to production. In CD, the software is built and delivered into production. In CI, different developers working on different modules upload the integrations frequently and reliably.

Ansible is a great tool for both CD and CI as it provides an infrastructure for setting the required environment and then deploying the application.

Now, let’s take a look at some more **Ansible interview questions** for freshers and experienced candidates.

## Q22. Explain Idempotency.

**Ans.**Idempotency is a feature of Ansible that ensures that only the required changes occur. One or more tasks can be executed multiple times, but it won’t change anything that is already been modified or is working correctly. It can be implemented in Ansible using the attribute created. For example, if a task is to create a directory on the server, then the directory will be created only if it does not already exist. Idempotency ensures a quality experience for both users and software teams.

**Also Read:**[**Top 8 Highest Paying IT Certifications**](https://www.naukri.com/learning/articles/top-paying-certifications/)

## Q23. What are the different strategies used to test Ansible projects?

**Ans.** There are three strategies to test Ansible projects:

1. **Manual Run:** It verifies that the system is in the desires state. While it is an easy way to test Ansible projects, it has increased risky because the results in a test environment might not be the same as in the production environment.
2. **Check Mode:**Check mode lets you know what the modules would have changed if the playbook was executed without check mode. It lets you check if the project behaves the way you want it to. Check mode is like a simulation and is the least used method in Ansible.
3. **Asserts:** It replicates how the test works in programming languages such as Python. It verifies the system has reached the actual state, not as a simulation, which you find in check mode. Asserts displays that the task did what it was expected to do.

## Q24. How to upgrade Ansible?

**Ans.** We can upgrade Ansible using the **sudo pip install ansible==<version-number>** command.

## Q25. Where are the Unit Tests available in Ansible?

**Ans.** Unit tests are available in .test/units/modules.

Let’s move on to some more **Ansible interview questions**.

## Q26. What are Ad-hoc commands?

**Ans.** Ad-hoc commands are one-line commands which can be run individually to perform a certain task quickly. These commands need not be performed later. They are an alternative to writing playbooks. While the playbooks are used for repeated action, ad-hoc commands are used when we want to perform a non-repetitive task. As ad-hoc commands are of one-time usage, they are not used for configuration management and deployment. They are best suited for tasks you repeat rarely.

An Ansible ad-hoc command uses the **/usr/bin/ansible** command-line tool.

***Explore popular***[***Online Programming Courses***](https://www.naukri.com/learning/programming-courses-certification-training-st619)

## Q27. What are the differences between Ansible and Puppet?

**Ans.** Below are some of the differences between Ansible and Puppet:

|  |  |
| --- | --- |
| **Ansible** | **Puppet** |
| Ansible is a provisioning, configuration, and deployment tool | It is more of a configuration management tool |
| The installation process is complex | Easy to install |
| Ansible uses YAML (Python) language | Has its own declarative language (Ruby) |
| GUI is less interactive than Puppet | GUI is more interactive and highly developed |
| More scalable than Puppet | Less scalable than Ansible |
| Less secure | More secure |

## Q28. How to increase the Ansible reboot module to more than 600 seconds?

**Ans.** The Ansible reboot module can be increased more than 600 seconds using the below syntax:

– name: Reboot a Linux system

  reboot:

    reboot\_timeout: 1000

## Q29. How to use docker modules in Ansible?

**Ans.** Docker modules require the Docker SDK for Python installed on the host running Ansible.

For Python 2.7 or Python 3, it can be installed using the following:

**$ pip install docker**

For Python 2.6, a version before 2.0 will be required. It can be installed using the following:

**$ pip install ‘docker-py>=1.7.0’**

## Q30. What are the differences between Ansible and Chef?

**Ans.** Below are the differences between Ansible and Chef:

|  |  |
| --- | --- |
| **Ansible** | **Chef** |
| Easy to set up. | Not very easy to set up compared to Ansible. |
| Ansible runs with a single active node (Primary instance). When the primary instance goes down, the Secondary instance fills in. | Chef uses a backup server that helps when the primary server goes down. |
| Uses YAML (Python) for managing configurations and is simple to understand. | Chef uses DSL (Ruby) for managing configurations. It has a steeper learning curve than YAML. |
| For Ansible, one has to pay $10,000 annually for 100 nodes. | It costs $13700 annually for 100 nodes. |
| Both Ansible and Chef are easily scalable. | |

## Q31. Explain the different types of Ansible Inventory.

**Ans.** There are two types of inventory files in Ansible:

**Static Inventory File:**

It is a plain text file containing a list of managed hosts declared under a host group using either hostnames or IP addresses. A host group name is enclosed in square brackets. The managed host entries are listed below the group name in each line.

Example:

[group name]

Host A ip\_address

Host B ip\_address

**Dynamic Inventory File:**

A dynamic inventory is a script written in [Python](https://www.naukri.com/learning/what-is-python-st619-tg21), PHP, or any [other programming language](https://www.naukri.com/learning/other-programming-languages-courses-certification-training-st619-tg1381). It is useful in cloud setups where IP addresses change once a virtual server is stopped and started again.

**Example:**

Create a demo\_aws\_ec2.yaml file for the config

**plugin: aws\_ec2 regions:**

**ap-south-1 filters:**

**tag:tagtype: testing**

We can fetch it using the below command:

**ansible-inventory -i demo\_aws\_ec2.yaml –graph**

**Also Read:**[**Top Universities Offering Free Online Programming Courses**](https://www.naukri.com/learning/articles/top-universities-offering-free-online-courses-for-programmers/)

## Q32. Explain Ad-hoc commands.

**Ans.** In Ansible, ad-hoc commands are used for performing a specific task. Ad-hoc commands are one-line commands that can be used as an alternative to writing playbooks.

## Q33. Define Ansible Galaxy.

**Ans.**Ansible Galaxy refers to the Galaxy website used to find, share, and download modules and roles by the Ansible users. This command-line tool is available with Ansible. It is used to install roles from Galaxy or the source management system. It can also be used to create new roles, remove old ones, and perform tasks on the Galaxy website. We can boost our project using Galaxy.

The syntax used to download roles from the Galaxy website:

**$ansible-galaxy install username.role\_name**

## Q34. How to use Ansible to create encrypted files?

**Ans.** Use the‘ ansible-vault create’ command to create encrypted files.

**$ ansible-vault create filename.yaml**

## Q35. What is an ask\_pass module?

**Ans.** Ask\_pass is a control module in an Ansible playbook that is used to control a password prompt when the playbook gets executed. By default, it is set to True.