Basic Ansible Interview Questions

1. What is CI/CD?

Continuous Integration is something that is used for streamlining the development and deployment process. These lead to the more rapid development of cohesive software.

Continuous Delivery is on the other hand is a process where your code after being pushed to a remote repository can be taken to production at any time.

In the above diagram our integration test and unit test are performed without any manual intervention and after UAT we just needed the approval to ship our tested features to production and to make such a process we need CI/CD.

2. 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.

You can download a PDF version of Ansible Interview Questions.

Download PDF

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.

33. Conclusion

Ansible is a great tool for automating IT tasks and it is widely used in industries, thus every software developer or someone in the DevOps team should know the basics. Also, it is very easy to set up so we can start right away. These questions cover the most important concepts related to Ansible which will help in both interview and understanding Ansible in depth.

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. Python Programming Course is one of the most demanding skills right now in the market.

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.

Go through this Ansible Cheat Sheet. This is a very useful handbook.

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

Check out this Ansible tutorial and practice for getting a better understanding of Ansible.

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.

Ansible CICD

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.

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.

Check our blog on What is An Ansible Playbook to get an in-depth understanding of Playbooks!

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.

Also, have a look at our DevOps course in Bengaluru if you are looking for a DevOps certification!

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 vs 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	

Career Transition

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 vs 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

Master the core concepts of Terraform through this Terraform Certification Course and become an expert!

Intermediate-level Ansible Interview Questions

Stepping up the level, let us now look into some intermediate-level Ansible interview questions and answers for experienced professionals:

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.

Learn from our blog Getting Started With Ansible Tower With A Hands-On

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 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

Even go through DevOps Interview Questions for better preparation.

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
```

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

40. Can docker modules be implemented in Ansible? If so, how can you use it?

Yes, you can implement docker modules in Ansible.

Ansible requires you to install docker-py on the host.

Command: $ pip install 'docker-py>=1.7.0'

The docker_service module also requires docker-compose.

Command: $ pip install 'docker-compose>=1.7.0'

41. How do you test Ansible projects?

There are three testing methods available:

Top 24 Ansible Interview Questions and Answers for 2023

Lesson 4 of 4By Sana Afreen

Last updated on Feb 14, 2023100368

Top 24 Ansible Interview Questions and Answers

Previous

Table of Contents

DevOps is attracting a lot of attention these days, which means anything associated with it also gets the spotlight, inviting increased interest and scrutiny. Ansible is a heavily favored DevOps tool, specifically in the realm of software automation. Consequently, it has a higher profile.

Since Ansible is increasing in popularity, it stands to reason that more businesses and organizations are looking for candidates who have experience using it. This is why we have curated the list of the most frequently asked Ansible interview questions and answers, ranging in level from beginner to advanced, which will help you prepare for your next interview seamlessly, and serve as a refresher.

Let us now begin by looking at some of the Ansible interview questions and answers for beginners.

Choose the Best for Your Career!

Caltech Program in DevOpsEXPLORE PROGRAMChoose the Best for Your Career!

Beginner Level Ansible Interview Questions For Freshers

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 and powered by Red Hat. It uses SSH to deploy SSH without incurring any downtime.

Related learning: Ansible for Beginners

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; 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, 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 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 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.

Next, let us look at the intermediate-level Ansible interview questions.

Learn Concepts - Basics to Advanced!

Caltech Program in DevOpsEXPLORE PROGRAMLearn Concepts - Basics to Advanced!

Intermediate Ansible Interview Questions

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

Unleash a High-paying Career in DevOps!

Caltech Program in DevOpsEXPLORE PROGRAMUnleash a High-paying Career in DevOps!

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 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, Salt, and of course, Ansible.

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

Kickstart Your DevOps Career With Us!

Caltech Program in DevOpsEXPLORE PROGRAMKickstart Your DevOps Career With Us!

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:"

Prepare to Answer all the Questions!

Caltech Program in DevOpsEXPLORE PROGRAMPrepare to Answer all the Questions!

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