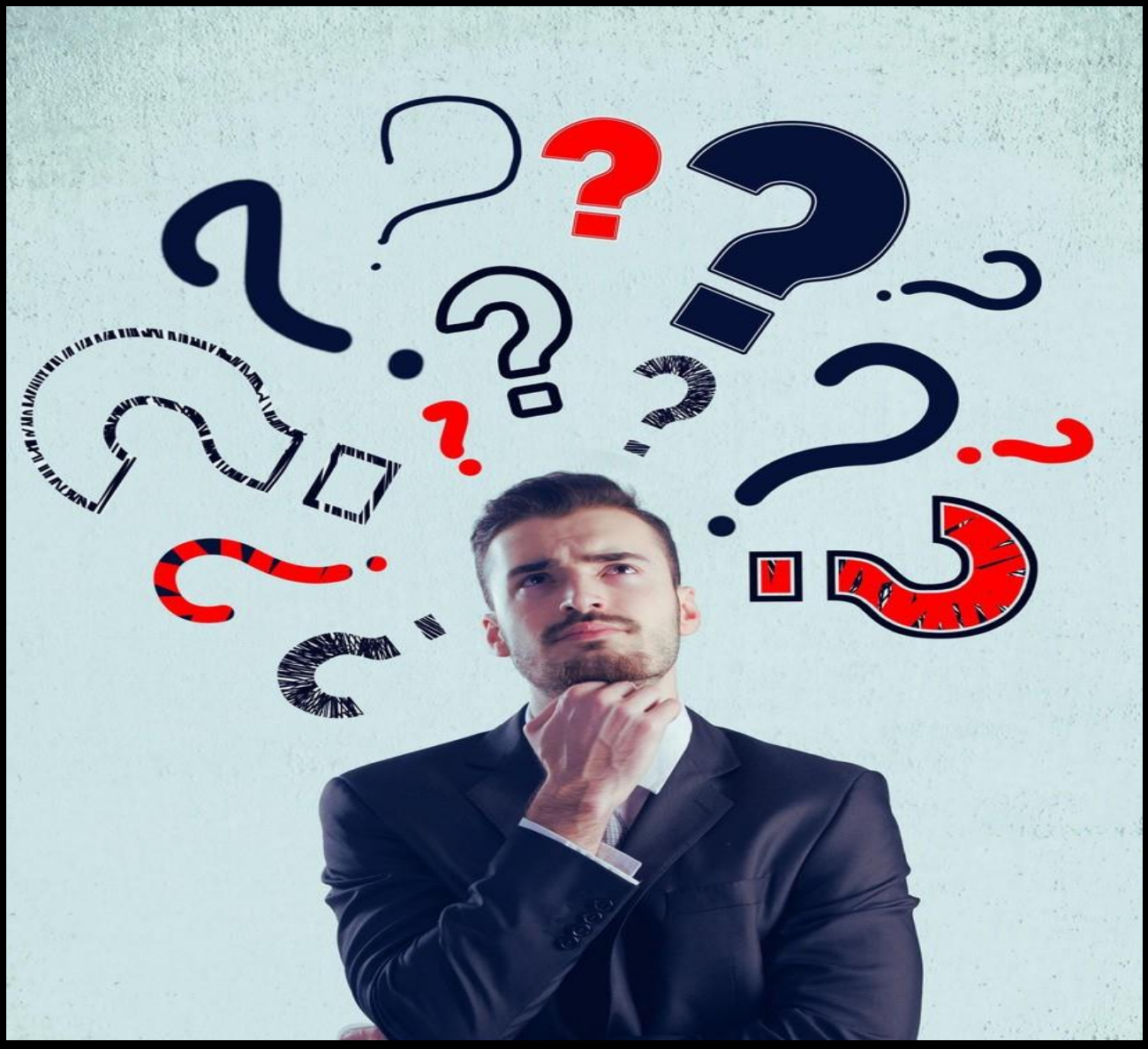
ANSIBLE

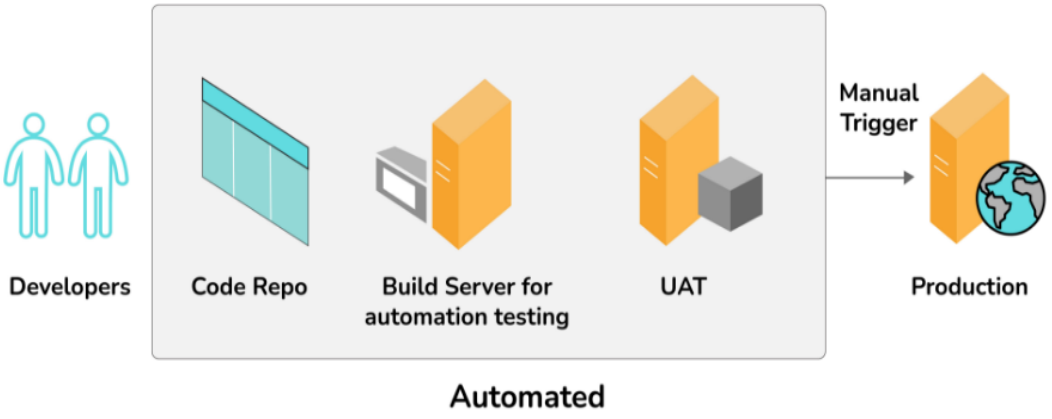
INTERVIEW QUESTIONS AND ANSWERS



Phone/WhatsApp: +1 (515) 309-7846 (USA)
Email: info@zarantech.com
Website: www.zarantech.com

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.

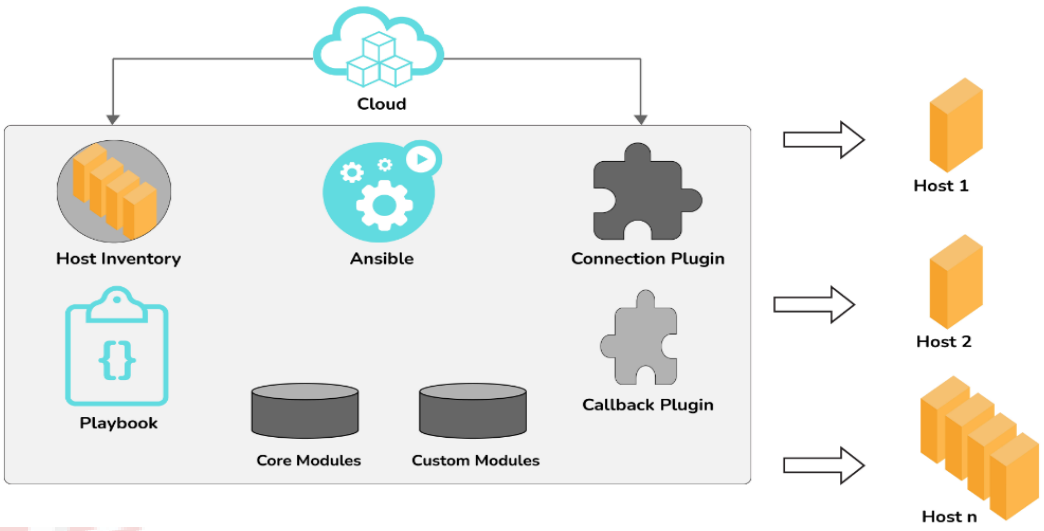
2. What is Configuration Management?

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.

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 – Password less 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 Open stack, Rack space, 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 in it, 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 impotently. The return value of these are JSON string in stout 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 are call-back plugins in Ansible?

Call-back 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 call back is used to record playbook events to a log file, and mail call back is used to send email on playbook failures. We can also add custom call back plugins by dropping them into a callback_plugins directory adjacent to play, inside a role, or by putting it in one of the call back directory sources configured in ansible.cfg.

9. 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.

10. What is Ansible Tower and what are its features?

Ansible Tower is an enterprise-level solution by Red Hat. 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.

11. 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.

12. 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.

13. 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.

14. 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 fulfils the requirements needed to achieve it.

15. 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.

16. 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.

17. 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.

18. 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.

19. How do you set up Ansible?

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

20. 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.

21. What is "idempotency"?

impotency is an important Ansible feature. It prevents unnecessary changes in the managed hosts. With impotency, 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.

22. 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.

23. 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.

24. 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.

25. 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.

26. 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.

27. 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 the need to install an agent on a remote system.

28. 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 this, we need some programming knowledge to understand the Ansible scripts.

29. What are the advantages of Ansible?

Ansible has many strengths, which include the following:

- It is agentless and only requires the SSH service running on the target machines.
- Python is the only required dependency, and, fortunately, most systems have Python 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 fulfils the requirements needed to achieve it.

30. What is configuration management?

Configuration Management (CM) is a practice that we should follow 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. CM keeps track of all updates that are needed in a system and ensures that the current design and build state of the system are up-to-date and properly functional.

31. What are the requirements for the Ansible server?

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

32. 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 as follows:

- 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 of organizing playbooks and other files 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.

33. What is a playbook?

A playbook has a series of YAML-based files that send commands to remote computers via scripts. Developers can configure completely 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.

34. 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.
{{hussars [inventory hostname] ['ansible_' + which interface] ['ipv4'] ['address']}}	# … vats: 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}}.

35. 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.

36. Which protocol does Ansible use to communicate with Linux and Windows?

In Linux systems, the Secure Shell (SSH) protocol is employed, while Windows systems utilize the Windows Remote Management (WinRM) protocol.

37. Compare Ansible with Chef.

Ansible	Chef
Easy to set up.	Not very easy to manage.
easy to manage.	Management is not easy.
The configuration language is YAML (Python).	The 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.

38. 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 into a playbook.

39. 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, which is stored in predefined variables, is available for use in the playbook. To generate facts, Ansible runs the set-up module.

40. When should you test playbooks and roles?

In Ansible, tests can be added to both new and existing playbooks. Therefore, most testing jobs offer clean hosting each time we use them. Using this testing methodology, we only need to make very minimal or zero code changes.

41. 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

42. 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 impotently. Their return value is JSON strings in stout and its input depends on the type of module.

The two types of modules are discussed below:

- Core Modules: These are modules that the core Ansible team maintains, and they will always ship with Ansible itself. The issues reported are fixed at a higher 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, they are being shipped with Ansible, but they might get discontinued in the future. Popular extra modules may be promoted to core modules over time. The source for these modules is hosted by Ansible on GitHub in Ansible-modules-extras.

43. What are call back plug-ins in Ansible?

call back plug-ins mostly control the output we see while running CMD (Command Prompt) programs. Apart from this, it can also be used for adding additional or, even, multiple outputs. For example, the log plays call back is used to record playbook events into a log file and the mail call back is used to send an email on playbook failures. You can also add custom call back plug-ins by dropping them into a callback_plugins directory adjacent to play, inside a role, or by putting them in one of the call back directory sources configured in ansible.cfg.

44. Define 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, namely static and dynamic, which are explained below:

- 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, any other programming language, or, preferably, using plug-ins. In a cloud setup, static inventory file configuration will fail since IP addresses change once a virtual server is stopped and started again.

45. 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.

46. 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.

47. What do you understand by the term 'idempotency'?

Idempotency is an important Ansible feature. It prevents unnecessary changes to 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, not the ones already in place.

48. 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.

49. What are the advantages of using Ansible?

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

- Agentless
- Very low overhead
- Good performance.

50. 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

51. What's the Use of Ansible?

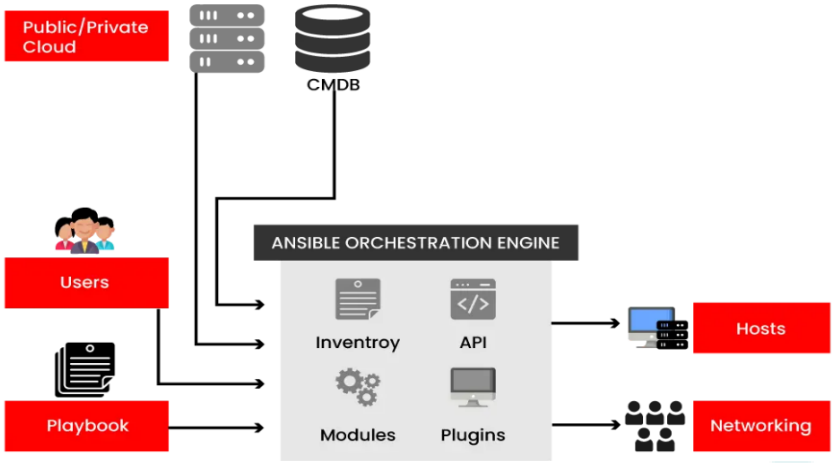
Ansible can be used in IT Infrastructure to manage and deploy software applications to remote nodes. For example, let's say you need to deploy a single software or multiple software to 100's of nodes by a single command, here ansible comes into the picture, with the help of Ansible you can deploy as 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.

52. How does Ansible Works?

- There are many similar automation tools available like 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 executing for Ansible when it's not managing any nodes.
- Ansible can handle 100's 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 are 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.

53. 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 are 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, Red Hat, 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.

54. 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 provides a stable infrastructure to a provision target environment and then deploys the application to it.

55. 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.

56. 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.

57. 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.

58. 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.

- 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.
- State - For creating a new file, this parameter should be set to touch.

59. Why are you attracted to science and science fiction?

Early imprinting, maybe, for science fiction. When I was quite small a family friend let me read his 1950s run of 'Galaxy' magazine. My favourite aunt pressed John Wyndham's 'The Day of the Traffics' 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 Krasner's and Newman's 'Mathematics and the Imagination' and all those books of Martin Gardner's 'Scientific American' columns.

60. 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 'Super Writer' 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 Super Writer 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 Loco script 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/.)

61. 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 favour 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.

62. 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.

63. 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:
  - Controlling machines
  - 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 that are capable of performing multiple tasks and they are in YAML file format.

64. What does Ansible offer?

Ansible offers:

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

65. 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.

66. 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.

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

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:

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

68. 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.

69. 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.

70. What is an Ansible role?

When you answer such Ansible interview questions, make sure to not go into details and answer it as an Ansible role is a pre-defined set of tasks, variables, and handlers organized in a structured way. Roles provide a modular approach to organizing and reusing Ansible code. They allow for easy sharing and reuse of common configurations, making it simpler to manage and maintain complex infrastructure.

71. How does Ansible differ from other configuration management tools?

Ansible differentiates itself from other configuration management tools by its agentless nature, simplicity, and ease of use. Unlike tools like Puppet or Chef, Ansible does not require any agents to be installed on remote hosts. It uses SSH or WinRM for communication, making it lightweight and easy to set up.

72. How can you handle variables in Ansible?

Ansible allows the use of variables to make playbooks more flexible and reusable. Variables can be defined at different levels, including host variables, group variables, and playbook variables. Variables can be assigned values inline or stored in separate variable files or inventories.

73. How do you handle error handling and retries in Ansible?

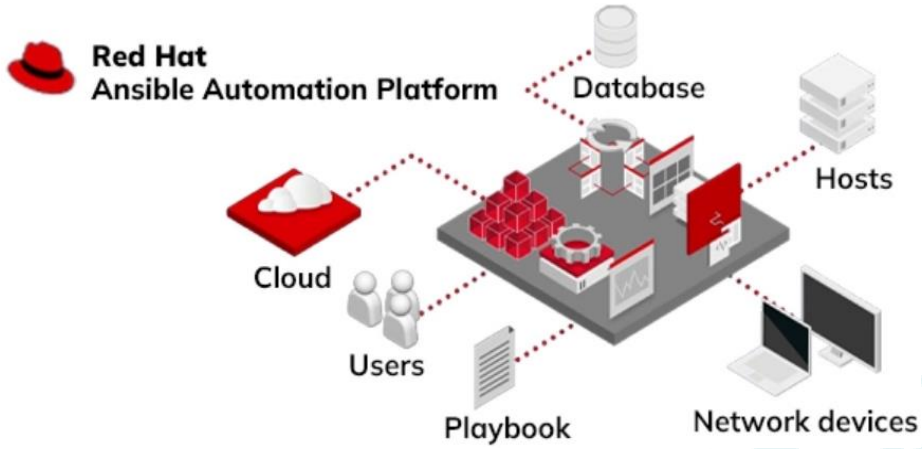
Ansible provides various error-handling mechanisms. You can use the "failed when" statement to specify conditions that determine task failure. Additionally, you can use the "ignore errors" option to ignore errors and continue with the playbook execution. For retries, you can use the "until" and "retries" parameters in tasks to retry a task until a certain condition is met. These are just a few basic questions and answers related to Ansible.

74. What is Ansible Fact?

Ansible Facts are system properties and variables automatically discovered and collected by Ansible. They provide information about the target hosts, such as network interfaces, operating system details, hardware information, and custom facts. Facts can be used in playbooks to make dynamic decisions based on the characteristics of the target hosts.

75. What is Red Hat Ansible?

Red Hat Ansible, encompassing both Ansible and Ansible Tower, stands as a comprehensive end-to-end automation platform. These platforms offer a range of features and functionalities, including:



- Provisioning: Setting up and preparing the necessary resources and infrastructure.
- Deploying Applications: Effortlessly deploying applications on the intended systems.
- Orchestrating Workflows: Streamlining and coordinating complex workflows to ensure seamless operations.
- Managing IT Systems: Overseeing and controlling various components of IT systems.
- Configuration of IT Systems: Fine-tuning and configuring IT systems to meet specific requirements.
- Networks: Managing and configuring networks efficiently.
- Applications: Handling the deployment and management of applications with ease.

76. How can you handle secrets and sensitive data in Ansible?

This might sound like a tough or complicated Ansible interview question, but this can simply be answered as: - Ansible provides a feature called "Ansible Vault" for securely storing and managing sensitive data such as passwords, API keys, and SSH private keys. Vault encrypts the sensitive data and decrypts it during playbook execution. The encrypted data is stored in an encrypted file, and the decryption key can be provided interactively or through automation.

77. Explain the difference between "handlers" and "tasks" in Ansible.

In Ansible, tasks represent a set of actions to be performed on a target host. They can include commands, module invocations, or file operations. Tasks are executed in the order they appear in a playbook. Handlers, on the other hand, are special tasks that are only executed when notified by other tasks. They are typically used to restart services or perform actions that are dependent on changes made by previous tasks.

78. What is an Ansible inventory?

Ansible inventory is a file that defines the hosts and groups of hosts that Ansible manages. It contains information such as IP addresses, hostnames, and groupings of hosts. The inventory file can be a simple text file or a dynamic inventory script that fetches host information from external sources like cloud providers or databases.

79. Can you explain the concept of idempotence in Ansible?

This question takes a prominent place among interview questions on Ansible. Idempotence in Ansible means that executing a task multiple times should have the same result as executing it once. Ansible ensures idempotence by checking the current state of the system against the desired state described in playbooks. If a task has already been completed successfully, Ansible skips it during subsequent runs, preventing unnecessary changes.

80. How can you test Ansible playbooks?

Ansible provides a tool called "ansible-playbook" that allows you to test playbooks. You can use the "--syntax-check" option to check the syntax of the playbook without executing it. The "--check" option performs a dry-run of the playbook, showing the changes that would be made without actually applying them. Additionally, you can use Ansible's "ansible-lint" tool to check for best practices and potential issues in your playbooks.

81. How does Ansible handle complex deployments involving multiple servers?

Ansible uses the concept of inventory and groups to handle complex deployments involving multiple servers. You can define groups in the inventory file and specify which tasks or playbooks should be executed on specific groups or hosts. Ansible also supports parallel execution, allowing tasks to run concurrently on multiple servers.

82. Can you explain the difference between Ansible ad-hoc commands and playbooks?

Ansible ad-hoc commands are used for executing simple tasks on remote hosts without the need for a playbook. Ad-hoc commands are useful for one-time or quick tasks. Playbooks, on the other hand, are YAML-based files that allow you to define more complex tasks and orchestrate multiple steps. Playbooks provide more flexibility, reusability, and maintainability compared to ad-hoc commands. This is an important Ansible interview question, make sure to explain this in detail a bit.

83. How can you handle the conditional execution of tasks in Ansible?

Ansible provides conditional statements such as "when" to control the execution of tasks based on specific conditions. You can define conditions using Jinja2 templating syntax to evaluate variables, facts, or other expressions. Tasks with conditions will only be executed if the condition evaluates to true.

84. What is Ansible Tower (AWX)?

Ansible Tower, now known as AWX, is an open-source web-based interface and automation platform built on top of Ansible. AWX provides additional features like a graphical dashboard, role-based access control, job scheduling, and a RESTful API. It allows for centralized management and monitoring of Ansible playbooks and provides a more user-friendly interface for managing automation tasks.

85. How can you extend Ansible's functionality using custom modules?

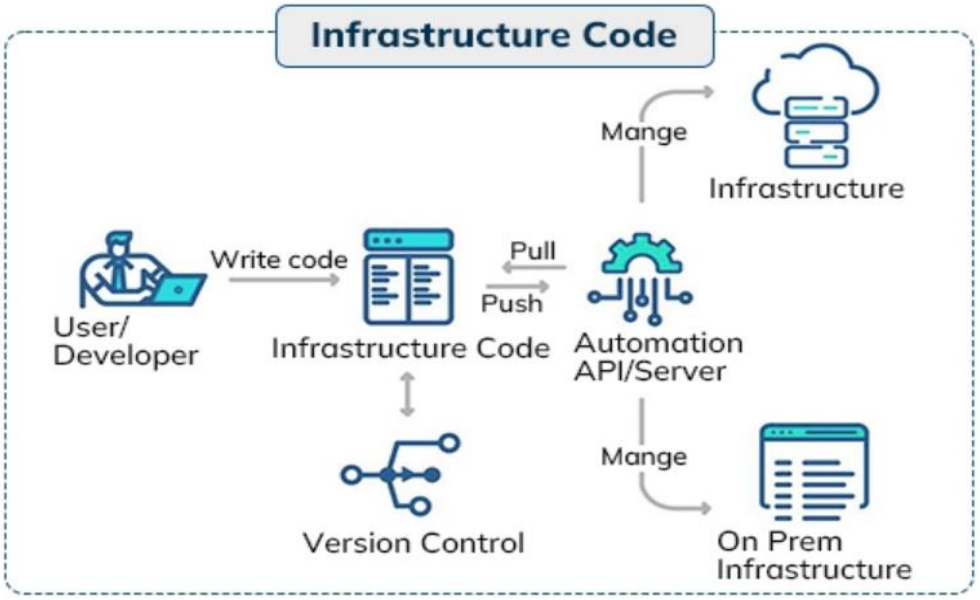
Ansible allows you to create custom modules written in programming languages like Python. Custom modules enable you to extend Ansible's functionality by performing specific tasks that are not covered by the built-in modules. Custom modules can be used in playbooks like any other module, allowing for greater flexibility and customization.

86. What is Configuration Management?

Configuration Management is a fundamental practice designed to help us keep a detailed record of all updates made to a system over time. This becomes particularly beneficial in situations where a significant bug emerges due to recent changes, and we need to address it swiftly with minimal disruption. Instead of immediately fixing the bug, Configuration Management empowers us to roll back the recent changes that led to the issue, thanks to our meticulous tracking. This proactive approach not only ensures system stability but also streamlines the process of resolving issues effectively.

87. Explain Infrastructure as Code.

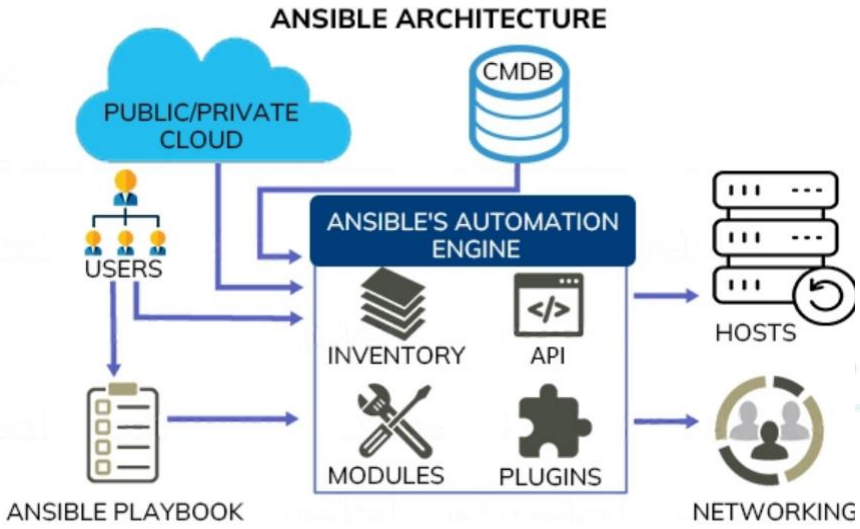
Infrastructure as Code (IaC) is a vital process embraced by DevOps teams to bring greater organization to infrastructure management. Rather than relying on ad-hoc scripts or manual configurations for cloud components, IaC promotes the practice of storing all configurations in a code repository.



Any adjustments to the configuration are made through this repository, which is also placed under source control for added wisdom. This approach enhances speed, consistency, and accountability in managing infrastructure, offering a more efficient and traceable way of handling changes.

88. Could you walk me through the Ansible architecture?

Absolutely. At the heart of Ansible is its automation engine, a pivotal player that collaborates directly with a variety of cloud services, the Configuration Management Database (CMDB), and individuals crafting playbooks to set the Ansible Automation engine in motion.



The Ansible Automation engine is composed of several key components:

- Inventories: Think of these as organized lists containing nodes, each with its unique IP address, servers, databases, and other entities requiring management.
- APIs: Just like any other API, Ansible's APIs facilitate seamless communication with a range of cloud services, whether they're public or private.
- Modules: These are essential tools for handling system resources, packages, libraries, files, and more. Ansible offers an extensive library of approximately 450 modules, enabling the automation of a broad spectrum of tasks.
- Plugins: When executing Ansible tasks as a job, Plugins come into play. They streamline task execution by creating a job-like environment containing code snippets tailored to specific functionalities. Ansible boasts a wealth of Plugins, including the Action plugin, serving as a user-friendly interface to modules and capable of executing tasks on the controller before calling the modules.

- Networking: Beyond system automation, Ansible extends its reach to automate diverse networks and services. This is achieved through playbooks or Ansible roles that seamlessly span different network hardware.
- Hosts: Referring to machines (Linux, Windows, etc.), Ansible Hosts/Node systems are the targets for automation.
- Playbooks: These are essentially simple code files outlining the tasks to be executed. Written in YAML format, Playbooks serve as versatile tools for automating tasks and declaring configurations.
- CMDB (Configuration Management Database): Acting as a repository for various IT installations, the CMDB stores data about IT assets, also known as configuration items (CIs), and outlines the relationships between these assets.
- Cloud: Ansible seamlessly integrates with the cloud—a network of remote servers on the Internet designed for storing, managing, and processing data, as opposed to a local server.

Such particular Ansible interview questions hold significance; and provide a detailed explanation for maximum impact.

89. Explain Ansible modules in detail.

Ansible modules serve as functional units or standalone scripts within the Ansible framework, executing specific tasks idempotently. These modules produce a JSON string in stdout as their return value, with input requirements varying based on the module type. They are employed by Ansible playbooks to automate and orchestrate tasks. Ansible modules are categorized into two types:

- Core Modules: Maintained by the core Ansible team, these modules are bundled with Ansible itself. Issues reported for core modules are prioritized for swift resolution, distinguishing them from those in the "extras" repository.
- Extras Modules: Managed by the Ansible community, these modules are currently shipped with Ansible but may be discontinued in the future. While still usable, feature requests or reported issues for extra modules may see lower-priority updates.

Notably, popular extra modules have the potential to transition into core modules over time. Separate repositories, namely ansible-modules-core and ansible-modules-extra, house these respective modules.

90. What does the term 'idempotency' mean?

Idempotency is a key feature in Ansible that ensures we avoid unnecessary alterations to managed hosts. In simpler terms, it allows us to run one or more tasks on a server as many times as necessary without impacting anything that is already modified and functioning correctly. To put it plainly, idempotency only introduces the changes that are required, steering clear of any modifications that are already in effect. This way, Ansible ensures efficiency by executing tasks without causing unnecessary disruptions to a system that's already working as intended.

91. What are the Ansible Server requirements?

Meeting the Ansible server requirements involves a couple of key elements:

- Operating System: If you're a Windows user, you'll need a virtual machine with Linux installed. This serves as the environment where Ansible operates seamlessly.
- Python Version: Ansible requires Python 2.6 or a newer version to be in place. This ensures compatibility and smooth functioning. If Python isn't already on your system, you might need to install it to meet this requirement.

92. What is Ansible, and why it stands out from other configuration management tools?

Ansible is a configuration management and automation tool powered by Red Hat and written in Python. Without installing an agent on the remote machine, users may utilize SSH to deploy and update software.

Below are some of Ansible features:

- Simple: Ansible uses a human-readable YAML syntax that is easy to write and understand.
- Agentless: Ansible does not need any agents or daemons running on the target machines.
- Efficient: Ansible reduces the time and effort required to perform IT tasks.

93. What are Ansible modules?

Ansible modules are similar to functions or standalone scripts that execute different operations in an idempotent manner. The output of these functions consists of JSON strings that are displayed in the standard output (stdout). The specific input required by each module varies depending on its type. It is mainly used by Ansible playbooks.

In Ansible, there are two types of modules. These are:

- Core Modules: These modules are directly maintained by the core Ansible team.
- Extras Modules: These modules are maintained by the Ansible community.

94. What are the basic components of Ansible architecture?

Components of Ansible architecture are:

- Modules: The modules manage the libraries, folders, system resources, packages, and other components. Numerous tasks may be automated with the help of Ansible modules.
- Inventories: An inventory refers to an initialization file containing specific information regarding several servers, databases, and IP addresses.
- APIs: These are the interfaces that Ansible mainly uses to communicate with the hosts and execute modules.
- Plugins: These are simply code that's main purpose is to extend or modify Ansible's functionality.
- Playbooks: It is a code file that has been written using the YAML format. A playbook outlines the specific tasks that must be executed or automated.
- Hosts: Hosts refer to the node system that is managed and configured by Ansible. It can run on any OS that supports APIs, the one that Ansible uses.
- Cloud: Instead of using a single server in a single location, In the cloud, data is distributed over a network of remote servers on the Internet.
- CMDB: CMDB stands for Configuration Management Database, and it is a repository of data that describes the assets and relationships in an IT environment.

95. What is a handler in Ansible?

Similar to standard playbook tasks, handlers are only executed when notified by another task. The main. yam file under roles/role name>/handlers are where the handlers are loaded automatically. The handlers mentioned may be accessed by any task associated with the given role and by any tasks associated with other roles that have listed the given role as a dependency. The handlers are executed only once after all tasks are executed in a specific play. In the event of a failure of the play on a specific host or all hosts before the handlers are alerted or notified, the handlers will not be executed unless handlers are explicitly forced to run using the command line flag –force-handlers during playbook execution.

96. Name Ansible tools

Here's a list of different tools you can mention during an Ansible developer interview:

- Ansible Galaxy: The Galaxy website allows you to find and share community roles to help save you time.
- Ansible Tower: a command-line tool (also with a graphical component) that provides additional functionality.
- Visual studio code: A Microsoft-developed code editor with support with YAML and Ansible through extensions.
- Atom: an open-source text editor for tracking project changes helpful in integrating with YAML files.

97. Explain the key Ansible terms

Here are a few basic terms you should know regarding Ansible:

- Controller machine: The place where you install Ansible. It provisions managed servers.
- Inventory: The initialization file with information about different servers you manage.
- Playbook: A code file written in YAML format. It has tasks the program needs to execute or automate.
- Task: A single procedure that the program needs to execute (e.g., installing a library).
- Module: A set of tasks the program can execute. Ansible has many built-in modules. You can also create custom ones.
- Role: A predefined way to organize playbooks and files to enable sharing and reusing portions of provisioning.
- Play: An entire task from start to finish. This is the execution of a playbook.
- Facts: Global variables that store system details. These include network interfaces or operating systems.
- Handlers: They trigger the status of a service, such as restarting or stopping.

98. What do you mean by 'facts' in Ansible?

When we execute an Ansible playbook, it first collects information ('facts,' a comprehensive list of all the environment characteristics) about each host in the play. Information gathering during playbook execution, such as host IP address, CPU type, disk space, OS information, and network interface information, can be used to modify the timing of tasks or the values used in configuration files. The 'ansible mining -m setup' command can be utilized with the setup module to obtain a comprehensive list of all gathered facts. Facts can be manually set using the set fact module or by passing variables to the ansible-playbook command.

99. How do you test Ansible projects?

Testing Ansible projects is one of the crucial steps to maintain the quality and reliability of your automation. Below are three types of testing methods for Ansible projects.

- Asserts: It duplicates the test execution process in other programming languages like Python. One can use it to check the state of a system, the output of a command, and a variable's value. It determines whether a task has effectively carried out the intended function and made the necessary modifications to the relevant resources.
- Check mode: In this mode, one can run Ansible projects without making any changes to target hosts. The roles and playbooks' scripts and commands cannot be executed in check mode. So, one has to disable the check mode by using the command check mode: no
- Manual run: It is simply running your Ansible projects on testing environments prior to deployment. A manual run is useful for testing the functionality and performance of your Ansible project and for catching any errors or bugs that might occur.

100. How does the Ansible firewalld module work?

The Ansible firewalld module is utilized for managing firewall rules on host machines. The functionality of this system is similar to that of the Linux firewalld daemon, as it gives the freedom to the user to allow or block services from the port.

Two main concepts of firewalld module are:

- Zones: This is the location where we can specify which services are accessible to a location that is connected to the local network interface.
- Services: A service is a predefined set of ports and protocols that can be allowed or denied by the firewall.

Corporate Training Course Catalog
https://bit.ly/devops-course-catalog

DevOps Learner Community
https://www.linkedin.com/showcase/devops-learner-community/

Get any DevOps Video Training
https://zarantech.teachable.com/courses/category/devops

Phone/WhatsApp: +1 (515) 309-7846 (USA)
Email: info@zarantech.com
Website: www.zarantech.com