**You said:**

ansible interview questions basic to advacne with solution

**ChatGPT said:**

Here's a list of **Ansible Interview Questions** ranging from basic to advanced, along with answers for each:

**Basic Ansible Interview Questions**

**1. What is Ansible?**

**Answer:**  
Ansible is an open-source automation tool used for c**onfiguration management, application deployment, and task automation**. It works by managing nodes over SSH (Linux) or WinRM (Windows), using simple YAML-based playbooks.

**2. What are Ansible Playbooks?**

**Answer:**  
Ansible Playbooks are **YAML files that describe the automation tasks** you want to run on remote systems. They **define the configuration, deployment, and orchestration tasks**.

**3. Explain Ansible Inventory.**

**Answer:**  
Ansible Inventory is a file where you define the hosts or machines on which Ansible will perform tasks. It can be a static file (/etc/ansible/hosts) or dynamic (generated on the fly using scripts).

Example:

ini

CopyEdit

[web]

web1.example.com

web2.example.com

[db]

db1.example.com

**4. What is a Role in Ansible?**

**Answer:**  
Roles are a **way of organizing tasks, files, templates, variables, and other** components into a reusable structure. **Each role is a folder that contains predefined tasks, defaults, handlers**, and other components.

**5. What is the purpose of ansible command?**

**Answer:**  
The ansible command is **used to run ad-hoc commands on remote** hosts. It allows you to execute tasks without needing to create a playbook.

Example:

bash

CopyEdit

ansible all -m ping

This command **sends a ping to all hosts in the inventory** to check connectivity.

**Intermediate Ansible Interview Questions**

**6. What is the difference between ansible and ansible-playbook?**

**Answer:**

* ansible is used to **run ad-hoc commands**.
* ansible-playbook is **used to execute Ansible playbooks**, which **define a series of automation steps.**

**7. Explain Ansible Modules.**

**Answer:**  
Ansible Modules are **reusable units of work that define tasks** to be executed on managed nodes. Common modules include yum, service, copy, file, and git. **Modules are executed during a playbook run** and return results.

**8. What is the use of ansible-galaxy?**

**Answer:**  
ansible-galaxy is a command-line **tool for managing roles and collections** in Ansible. It is **used to download, share, and manage community-contributed roles and collections** from Ansible Galaxy.

**9. What is Ansible Vault?**

**Answer:**  
Ansible Vault is a tool **used to encrypt sensitive data such as passwords or keys** within Ansible playbooks. It helps **secure private information** in a way that it can still be versioned and shared safely.

Example:

bash

CopyEdit

ansible-vault create secrets.yml

**10. What is the purpose of notify in Ansible?**

**Answer:**  
notify is used **to trigger handlers**. Handlers are **special tasks that are only executed when notified by another task.** Typically, they are used for actions like restarting services after a configuration change.

**Advanced Ansible Interview Questions**

**11. What are Ansible Fact Gathering and How Does it Work?**

**Answer:**  
Ansible gathers facts about the managed nodes (e.g., OS, IP address, memory) using the setup module by default. These facts are stored in variables and can be used throughout the playbook.

To disable fact gathering:

yaml

CopyEdit

gather\_facts: no

**12. What is Ansible Tower?**

**Answer:**  
Ansible Tower is a **web-based user interface and dashboard for managing** Ansible automation. It provides a central place to control and monitor automation tasks, manage inventories, and schedule jobs.

**13. What is the difference between include and import in Ansible?**

**Answer:**

* include: This is **used to dynamically include files**. The **included file can be executed at runtime** and is evaluated at the time the playbook is run.
* import: This is **used to statically include files during playbook parsing**. The tasks or roles are loaded before the playbook runs.

**14. How do you handle errors in Ansible?**

**Answer:**  
Ansible provides several ways to handle errors:

* **ignore\_errors: yes**: **Skips the task if** it fails.
* **block/rescue/always**: **Provides a way to handle errors and recovery tasks** within a block of tasks.
* **failed\_when**: Customizes the condition under which a task is marked as failed.

**15. What is Ansible Dynamic Inventory?**

**Answer:**  
Dynamic inventory **refers to generating the inventory dynamically using an external script** or plugin instead of using a static file. It is useful when managing large infrastructures or environments where hostnames change frequently.

Example:

bash

CopyEdit

ansible -i inventory.py all -m ping

**16. How does Ansible handle parallel execution?**

**Answer:**  
Ansible **runs tasks in parallel across multiple nodes**, based on the forks setting. You can configure the number of parallel processes using the -f flag in the command or in the configuration file (ansible.cfg).

Example:

bash

CopyEdit

ansible-playbook -i inventory site.yml -f 10

**17. What are Ansible Handlers?**

**Answer:**  
Handlers **are tasks that are only executed when notified by another** task. They are usually **used for tasks that need to be triggered by changes**, such as restarting a service after a configuration change.

Example:

yaml

CopyEdit

tasks:

- name: Update configuration file

copy:

src: config.txt

dest: /etc/app/config.txt

notify:

- restart service

handlers:

- name: restart service

service:

name: app

state: restarted

**18. What is ansible-pull?**

**Answer:**  
ansible-pull is a command that allows anode to **pull playbooks from a Git repository and execute them locally.** It is useful for distributed systems where each node is responsible for pulling its configuration.

Example:

bash

CopyEdit

ansible-pull -U https://github.com/myorg/repo.git

**19. How can you increase the performance of Ansible?**

**Answer:**  
Some tips for improving Ansible performance:

* Use async and poll **to run tasks asynchronously**.
* Set forks to increase parallelism.
* Use gather\_facts: no when you don't need facts for tasks.
* Avoid unnecessary task retries.
* Use cache for fact storage.

**20. How do you write Idempotent Playbooks in Ansible?**

**Answer:**  
Ansible is idempotent by default, meaning **running the same playbook multiple times will produce the same result.** To ensure idempotence:

* Always **use the state parameter for tasks** (e.g., state: present or state: absent for files, packages, services).
* Ensure that tasks check if changes are needed before making them (e.g., creates, regexp).

4o

**You said:**

contineu

**ChatGPT said:**

Certainly! Here are more advanced Ansible interview questions and their solutions:

**Advanced Ansible Interview Questions (Continued)**

**21. What are Ansible Collections?**

**Answer:**  
Ansible **Collections are a way to package and distribute automation content**, including roles, modules, and plugins. They are a more structured way to organize Ansible content compared to traditional roles.

* **Collections allow grouping of related roles, plugins, and modules.**
* They can be shared via Ansible Galaxy, improving content reuse and collaboration.

Example:

bash

CopyEdit

ansible-galaxy collection install community.general

**22. What is the difference between state: started and state: running in Ansible?**

**Answer:**

* state: started e**nsures that a service is started, but it does not check whether it is already running.** If it is stopped, Ansible will start it.
* state: running is used when you want to ensure that the **service is both started and running.**

Example:

yaml

CopyEdit

service:

name: nginx

state: running # **Ensures the service is not only started but also running**.

**23. Explain Ansible’s delegate\_to feature.**

**Answer:**  
delegate\_to is used **to run a task on a different machine than the one defined in the inventory**, which is helpful when a task needs to be run from a central server but affects other systems.

Example:

yaml

CopyEdit

- name: Copy file to web server

copy:

src: /tmp/file.txt

dest: /var/www/html/

delegate\_to: web1.example.com

**24. What is the ansible.cfg file and its use?**

**Answer:**  
ansible.cfg is the **configuration file used to define settings that control Ansible’s behavior**. It can be set in several places (global, user-level, or project-level) and is used for defining:

* Default inventory location
* **Remote user**
* **SSH connection parameters**
* Forks and parallel execution settings
* Logging settings

Example:

ini

CopyEdit

[defaults]

inventory = ./inventory

remote\_user = ubuntu

forks = 5

**25. How do you debug playbooks in Ansible?**

**Answer:**  
To debug playbooks in Ansible, you can use the following techniques:

* Use -vvv (increased verbosity) to **get more detailed output**.
* Use debug module to print variables or values at different stages of playbook execution.
* Set ANSIBLE\_DEBUG=true to **enable debug logs.**

Example:

yaml

CopyEdit

- name: Debug variable output

debug:

msg: "{{ variable\_name }}"

**26. What is the role of when in Ansible?**

**Answer:**  
The when statement in Ansible is **used to conditionally execute tasks or blocks of tasks based on the evaluation of a condition**. This is useful for skipping tasks that are not needed under certain conditions.

Example:

yaml

CopyEdit

- name: Install package only on Debian

apt:

name: "{{ item }}"

state: present

with\_items:

- apache2

- nginx

when: ansible\_facts['distribution'] == "Debian"

**27. Explain loop and with\_items in Ansible.**

**Answer:**

* with\_items is a legacy way of looping through a list in Ansible. It has now been replaced by the loop keyword for better clarity and flexibility.
* loop is the preferred way to iterate over a list, dictionary, or other iterable structures.

Example with loop:

yaml

CopyEdit

- name: Install multiple packages

package:

name: "{{ item }}"

state: present

loop:

- nginx

- apache2

**28. What are Ansible Callbacks?**

**Answer:**  
**Callbacks are used to hook into different points** of Ansible’s execution and customize the behavior during a playbook run. You can use them to create custom output, log to external systems, or manage failures.

Example:

bash

CopyEdit

ANSIBLE\_STDOUT\_CALLBACK=json ansible-playbook site.yml

This will produce JSON output instead of the default text output.

**29. How do you handle dependencies between roles in Ansible?**

**Answer:**  
To handle role dependencies in Ansible, **you can define the dependencies within a role’s metadata file (meta/main.yml).** This ensures that the required roles are executed in the correct order.

Example (meta/main.yml):

yaml

CopyEdit

dependencies:

- role: common

- role: apache

**30. How does Ansible handle configuration drift?**

**Answer:**  
Ansible **ensures that systems are in a known desired** state, thus avoiding configuration drift. If a system’s state changes from the desired configuration, Ansible will remediate the changes on the next run by applying the playbook again.

Example: If a configuration file is modified, Ansible will replace it with the correct version during the next playbook run.

**Specialized/Expert-Level Ansible Interview Questions**

**31. How can you scale Ansible to handle large infrastructures?**

**Answer:**  
To scale Ansible for large infrastructures, you can:

* Use **Ansible Tower** for **better management and scaling**.
* Leverage **parallelism** (with the forks setting) to run multiple tasks in parallel.
* Use **dynamic inventories** to automatically generate hosts for cloud environments (AWS, Azure, etc.).
* Implement **local caching** of facts using fact\_caching to reduce the overhead of gathering facts repeatedly.
* Use **Ansible Semaphore** or **AWX** for managing long-running jobs and complex workflows.

**32. Explain how Ansible integrates with Cloud Providers (AWS, GCP, Azure).**

**Answer:**  
Ansible integrate**s with cloud providers by using specific Ansible modules designed for each provide**r. These **modules can automate provisioning, management, and configuration of resources in the cloud.**

For AWS:

yaml

CopyEdit

- name: Create an EC2 instance

ec2:

key\_name: my\_key

region: us-west-2

image: ami-12345678

instance\_type: t2.micro

wait: yes

Similarly, you can use modules like gcp\_compute\_instance for Google Cloud and azure\_rm\_virtualmachine for Azure.

**33. What are Ansible’s strategies (linear, free, etc.)?**

**Answer:**  
Ansible provides different strategies to control how tasks are executed:

* **linear** (default): **Tasks are executed sequentially across all hosts**.
* **free**: Tasks are executed on all hosts simultaneously, without waiting for the previous task to finish.
* **host\_pinned**: Tasks are executed sequentially per host, but in parallel across hosts.
* **serial**: Limits how many hosts are processed at once (useful for rolling updates).

Example of serial:

yaml

CopyEdit

- hosts: webservers

serial: 3 # Run tasks on 3 webservers at a time

tasks:

- name: Update web server

yum:

name: httpd

state: latest

**34. What are the differences between Ansible and Puppet/Chef?**

**Answer:**

* **Language**: **Ansible uses YAML**, while Puppet and Chef use Ruby-based DSL.
* **Architecture**: Ansible **uses an agentless architectu**re (via SSH), whereas Puppet and Chef require **agents installed on the managed systems**.
* **Ease of Use**: Ansible is often considered easier to use due to its simple YAML syntax and declarative nature.
* **Scaling**: While all tools can scale, Puppet and Chef are typically used for large-scale enterprise environments, and Ansible can be scaled with tools like Tower or AWX.

**35. How do you handle complex workflows with Ansible?**

**Answer:**  
For complex workflows, Ansible supports:

* **Playbook include/import** for **modularizing large playbooks**.
* **Roles** for **reusable components**.
* **Blocks** for organizing tasks with conditional execution.
* **Handlers** to **trigger tasks after specific changes**.
* **Ansible Tower/AWX** to manage complex workflows, including scheduling and running playbooks in sequence.

**Conclusion**

These questions cover a broad range of Ansible concepts from basic usage to advanced configurations and scaling strategies. Interviewers use these to test candidates’ knowledge of both Ansible’s capabilities and how it integrates into a larger IT ecosystem. Understanding these concepts will give you a well-rounded ability to work with Ansible in various environments.