**Ansible Tutorials**

**What is Ansible?**

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

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

In simple words, Ansible acts as a command center for your IT infrastructure, enabling you to control and manage multiple servers, network devices, and cloud environments from a single location. It uses a declarative language called YAML to define the desired state of your infrastructure, specifying how systems should be configured and what tasks should be performed.

Instead of manually logging into each server or device and executing commands one by one, Ansible automates these tasks by connecting to the systems over SSH (Secure Shell) or other remote protocols. It allows you to define playbooks, which are sets of instructions that describe the desired configuration and actions to be taken on target systems.

**Why Ansible is used in DevOps?**Ansible helps automate the implementation of internally generated applications to your production programs to make DevOps easier

**Types of Configuration management tools :**

Chef - 2008

Puppet - 2005

Ansible - 2011

**Files and Directory are Present in Ansible Directory**

**1) ansible.cfg:**

**2) hosts:**

**3) roles:**

**Ansible Architecture**

The Ansible orchestration engine interacts with a user who is writing the Ansible playbook to execute the Ansible orchestration and interact along with the services of private or public cloud and configuration management database. You can show in the below diagram, such as:

**Inventory**

Inventory is lists of nodes or hosts having their IP addresses, databases, servers, etc. which are need to be managed.

**API's**

The Ansible API's works as the transport for the public or private cloud services.

**Modules**

Ansible connected the nodes and spread out the Ansible modules programs. Ansible executes the modules and removed after finished. These modules can reside on any machine; no database or servers are required here. You can work with the chose text editor or a terminal or version control system to keep track of the changes in the content.

**Plugins**

Plugins is a piece of code that expends the core functionality of Ansible. There are many useful plugins, and you also can write your own.

**Playbooks**

Playbooks consist of your written code, and they are written in YAML format, which describes the tasks and executes through the Ansible. Also, you can launch the tasks synchronously and asynchronously with playbooks.

**Hosts**

In the Ansible architecture, hosts are the node systems, which are automated by Ansible, and any machine such as RedHat, Linux, Windows, etc.

**Networking**

Ansible is used to automate different networks, and it uses the simple, secure, and powerful agentless automation framework for IT operations and development. It uses a type of data model which separated from the Ansible automation engine that spans the different hardware quite easily.

**Cloud**

A cloud is a network of remote servers on which you can store, manage, and process the data. These servers are hosted on the internet and storing the data remotely rather than the local server. It just launches the resources and instances on the cloud, connect them to the servers, and you have good knowledge of operating your tasks remotely.

**CMDB**

CMDB is a type of repository which acts as a data warehouse for the IT installations.

**Ansible ad-hoc Commands**

Ad-hoc commands are one of the simplest ways of using Ansible. These are used when you want to issue some commands on a server or bunch of servers. The ad-hoc commands are not stored for future use, but it represents a fast way to interact with the desired servers.

**Ansible Playbooks**

Playbooks are the files where the Ansible code is written. Playbooks are written in YAML format. **YAML** means "Yet Another Markup Language," so there is not much syntax needed. **Playbooks** are one of the core features of Ansible and tell Ansible what to execute, and it is used in complex scenarios. They offer increased flexibility.

Playbooks contain the steps which the user wants to execute on a particular machine. And playbooks are run sequentially. Playbooks are the building blocks for all the use cases of Ansible.

Ansible playbooks tend to be more configuration language than a programming language.

Through a playbook, you can designate specific roles to some of the hosts and other roles to other hosts. By doing this, you can orchestrate multiple servers in very different scenarios, all in one playbook.

**Ansible Roles**

Ansible roles is a concept that deals with ideas rather than events. Its basically another level of abstraction used to organize playbook.

They provide a skeleton for an independent and reusable collection of variables, tasks, templates, files, and modules which can be automatically loaded into the playbooks

Roles provide a framework for fully independent or interdependent collections of files, tasks, templates, variables, and modules.

The role is the primary mechanism for breaking a playbook into multiple files. This simplifies writing **complex playbooks** and makes them easier to reuse. The breaking of the playbook allows you to break the playbook into reusable components.

Each role is limited to a particular functionality or desired output, with all the necessary steps to provide that result either within the same role itself or in other roles listed as dependencies.

Roles are not playbooks. Roles are small functionality that can be used within the playbooks independently. Roles have no specific setting for which hosts the role will apply.

**Example:**

suppose you want your PlayBook to perform ten different tasks on five different systems would you use a single playbook for this well I would say no using a single playbook will make it more confusing and obviously it would be more prone to blunders so instead what you can do is you can create 10 different roles where each row will perform one task then all you need to do is mention the name of the role inside the play book to call them

[root@ansible-server test2]# tree

.

`-- role1

    |-- defaults

    |   `-- main.yml

| -- files

| ‘—min.yaml’

    |-- handlers

    |   `-- main.yml

    |-- meta

    |   `-- main.yml

    |-- README.md

    |-- tasks

    |   `-- main.yml

    |-- tests

    |   |-- inventory

    |   `-- test.yml

    `-- vars

        `-- main.yml

**Explanation**

The YAML file in the default directory contains a list of default variables that are to be used along with the playbook.

Contains files that can be deployed by this role. It contains files that need to be send to the hosts while configuring the role.

The handler's directory is used to store handlers.

The meta-directory is supposed to have information about the author and role dependencies.

The tasks directory is the main YAML file for the role. Contains the main lists of tasks that are to be executed by the role.

Template contains files which can be modified and added to the host being provisioned jinja2(template language) is used to achieve the modifications

The tests directory contains a sample YAML playbook file and a sample inventory file and is mostly used for testing purposes before creating the actual role.

The vars directory contains the YAML file in which all the variables used by the role will be defined. The directory templates and the directory files should contain files and templates that will be used by the tasks in the role.

**Ansible Galaxy**

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

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

Ansible roles consist of many playbooks, which is a way to group multiple tasks into one container to do the automation in a very effective manner with clean, directory structures.

**Ansible Variables**

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

**1) What is configuration management?**

Configuration management is a methos or it’s a process of managing your software system or hardware, and infrastructure components.

**What is Ansible?**

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

In simple words, Ansible acts as a command center for your IT infrastructure, enabling you to control and manage multiple servers, network devices, and cloud environments from a single location. It uses a declarative language called YAML to define the desired state of your infrastructure, specifying how systems should be configured and what tasks should be performed.

**2) Do you think ansible is better than other configuration management tools? If yes why?**

The preference for Ansible or other configuration management tools depends on specific use cases and requirements. Ansible is known for its simplicity, agentless architecture, and popularity among users. However, the choice between Ansible and other tools like Puppet, Chef, or SaltStack depends on factors like existing infrastructure, team expertise, and project needs. There is no one-size-fits-all answer.

**i)** Ansibe is agentless .

so what is agentless -> if you want to configure hundred of virtual machine or server so to configure all of this you don’t have to install any agent when you are using ansible as a configuration management tool

**ii)** ansible is written in python

**iii)** ansible is using a yaml file for wrirtting file

**3) Can you write an Ansible playbook to install Httpd services and get it running**

**4) How Ansible helped your organization**

Ansible has helped organizations automate configuration management, application deployment, and infrastructure provisioning, reducing manual errors and increasing efficiency. It simplifies collaboration, scales infrastructure, and accelerates tasks, resulting in cost savings and improved productivity.

so you can say that previously you're using some shell scripts as well as powershell scripts and some manual tasks to connect to these vms and make the required changes but once we adopted ansible we have reduced this time so answer will was very effective and it saved a lot of time

**5) What is Ansible dynamic inventory**

Ansible's dynamic inventory allows you to generate inventory information from external sources, such as cloud providers, databases, or scripts, making it suitable for dynamic environments.

**6) What is ansible tower and how you used it? If yes why?**

Ansible Tower is a web-based platform built on Ansible that provides a graphical interface, scheduling, role-based access control, and a dashboard for managing and monitoring Ansible automation.

**7) How do you manage the RBAC of users for ansible tower?**

RBAC in Ansible Tower is managed by defining organizations, teams, and users, and assigning permissions based on roles. This allows for granular control over who can access and modify automation workflows.

**8) What is ansible galaxy command and why is it used for?**

The "ansible-galaxy" command is used to interact with Ansible Galaxy, a platform for sharing and downloading Ansible roles. It simplifies the process of finding and integrating community-contributed roles into your playbooks.

**9) Can you explain me structure of ansible playbook using roles?**

An Ansible playbook using roles is organized into directories and files to modularize tasks. Roles contain tasks, templates, and variables specific to a component or function, promoting code reuse and playbook maintainability.

**10) What are handlers in ansible and why are they used?**

Handlers in Ansible are tasks that are triggered only when specific tasks make changes. They are often used to manage services or perform actions in response to changes initiated by tasks.

**11) I would you like to run a specific set of tasks only on windows vms and not linux vms is it possible?**

**Does ansible supports parallel execution of tasks?**

**What is protocol that ansible use to connect to windows vms?**

**Can you replace them in the order of predence?**

**Playbook group\_vars, role vars and extra vars**

**How do you handle secrets in ansible**

**Can we use ansible for iac? If yes, can you compare it with any other iac tools like terraform**

**Can you talk about a ansible playbook how it helped your company?**

**what do you think that ansible can improve?**