**Ansible Assignment 1**

1. What is the difference between CI and CD?

* CI CD are the widely used in devops
* Continuous integration
* Almost all Developers merge their changes into the main branch quite often because day in and day out the requirements will be changing as per business so app developers will have to change their code and once done they merge into main source code.
* Also in big organization there will be a bunch of developer who works on same project but in different aspects with different task so all team members will manage a common centralized repository where they check in and out there source code very often.
* Also they run lot of tests before merging and once the test is successful of the goal is achieved of the code then they will merge into source main code.
* They can also manage one main branch and inside that can be various sub branch for different teams with different task.
* Continuous integration puts a great emphasis on testing automation to check that the application is not broken whenever new commits are integrated into the main branch.
* Continuous delivery is an extension of continuous integration, with CD we can automatically deploys all source code changes to a testing and/or production environment after the build stage.
* For eg. after testing is done with help of automation we have an automated release process and we can deploy application any time by clicking a button.
* In continuous delivery we can decide to release daily, weekly, fortnightly, or whatever suits business requirements.
* Ideally we should deploy to production as early as possible to make sure that our release small batches that are easy to troubleshoot in case of a problem.
* Continuous deployment is one step further than continuous delivery. With this every change that passes all stages of production pipeline is released to our customers with no human intervention
* In short now one of the final difference is in continuous Delivery app deployment is manual by choosing a particular day while continuous development is fully automated. app deployment can be done as soon as testing is done and there is no release day.
* 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.

1. What is Configuration Management, and how does it work?

* Configuration management is 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 is help when there a real issue comes into production and we can revert that change with minimum downtime.
* Instead of fixing the issue we can roll back the new changes(which caused this issue or bug) as we have been tracking of those changes and previous state.
* so in short we can say configuration management is a process for maintaining computer systems, servers, and software in a desired, consistent state. It is a way to make sure that a system performs as it’s expected to as changes are made over time.
* Configuration Management is the practice of handling updates and changes systematically so that a system maintains its integrity over time.
* Configuration Management helps organization by overcoming the following challenges:
* Finding out what changes need to be implemented when user requirements change.
* Redoing and updating an implementation due to change in the requirements since the last implementation.
* Reverting to an older version of the component because the latest version is flawed.
* Replacing the wrong component because you couldn’t accurately determine which component needed replacing.
* With a real-time scenario there was an issue once in NYSE
* The New York Stock Exchange (NYSE) faced an issue in their software which prevented them from trading stocks for approx 90 minutes. On the previous night there was new software installed and that did not went as expected into production next day which create an outage so within those approx 90 min the change was reverted with help of this CM tools. which helped them to minimize the impact/downtime of prod servers.
* Configuration management works by writing scripts and code that will automate certain tasks to successfully configure all parts of a system together.
* few configuration management strategies are not fully automated and it will require manual intervention
* Configuration management tools collect information about the system and use it to determine details, variables and operations essential for the system to function at any given moment.
* Based on this information, the tools develop templates for expanding system operations, track behaviours across the system and identify the stage of update cycles or software operations.
* Configuration management supervises the behaviour of all parts of a network's infrastructure and informs the way they interact.

1. What is Ansible, and describe it’s working?

* Ansible is an open source automation platform that etc.
* Ansible is an open source automation platform. It is used for configuration management, application deployment task automation It is a simple automation language that can perfectly describe an IT application infrastructure in Ansible Playbooks. It is also an automation engine that runs Ansible Playbooks.
* Ansible can manage powerful automation tasks and can adapt to many different workflows and environments. At the same time, new users of Ansible can very quickly use it to become productive
* Ansible Is Simple
* Ansible Is Powerful
* Ansible Is Agentless
* In ansible there two types of servers – Controlling machines and Nodes. Controlling machine is where Ansible is installed and nodes are the ones that are managed by the controlling machines through SSH. There is an inventory file in the controlling machine that holds the location of the node systems. Ansible deploys modules on the node systems by running the playbook on the controlling machine. Ansible is agentless, that means there is no need to have a third party tool to make a connection between one node and the other.
* 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.
* In simple words for eg. there are 100 servers which are new servers to be build and we have a requirement to install a particular software like say Apache Kafka then with the help of Ansible and Ansible Tower we can easily push this software in just few simple steps to all those 100 machines without logging into those 100 machines. So that is a small eg. for Ansible in IT industry.

1. What distinguishes Ansible from other similar tools?

* Because of below features makes it different from other tools.For automation from the start and scratch, Ansible works better than other tools
* It is agentless or we can say No agent is required to push the code/software to target machine
* It is very easy and simple to install when compare with other tools
* It requires SSH service running on target machines.
* Python /Yaml is the only required dependency.
* It requires minimal resources, so there is low overhead.
* It is easy to learn and understand since Ansible tasks are written in YAML.
* most of other tools are procedural, Ansible is declarative, it defines the desired state and fulfils the requirements needed to achieve it.
* For Configuration Management – it is like push and pull
* scalability is very high
* it will streamlined provisioning
* It is secure and compliance integration into automation
* Availability: Ansible has backup secondary nodes
* Most modules are idempotent, which means that they can be run safely multiple times, and if the system is already in the correct state, they do nothing.

1. What is the purpose of the Ansible Galaxy?

* It is a public library of Ansible content written by a variety of Ansible administrators and users.
* It contains thousands of Ansible roles and it has a searchable database that helps Ansible users identify roles that might help them accomplish an administrative task.
* Ansible Galaxy includes links to documentation and videos for new Ansible users and role developers.
* 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.
* ansible-galaxy-collection command implements similar to init, build, install, etc like an ansible-galaxy command
* To create the base directory structure we can use a tool bundled with Ansible called as ansible-galaxy.

1. Can you go over the Ansible modules in detail?

* Ansible ships with hundreds of useful modules that can perform a wide variety of automation tasks.
* They can act on system files, install software, or make API calls.
* When used in a task, a module generally ensures that some particular aspect of the machine isin a particular state. For example, a task using one module might ensure that a file exists and has particular permissions and contents, while a task using a different module might make certain that a particular file system is mounted. If the system is not in that state, the task should put it inthat state. If the system is already in that state, it does nothing. If a task fails, the default Ansible

behaviour is to abort the rest of the playbook for the hosts that had a failure.

* Tasks are idempotent. This means that you can safely run a playbook on the same hosts multiple times. When your systems are in the correct state, the

playbook makes no changes when you run it. This means that you should be able to run a playbook on the same hosts multiple times safely

* The return value of these are JSON string in stdout and input depends on the type of module.
* These are used by Ansible playbooks. Ansible has hundreds of built-in modules but you can also create custom ones. 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.
* In core module 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.
* these are being shipped with Ansible but chances are there 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.
* The source for these modules is hosted by Ansible on GitHub in Ansible-modules-extras.
* 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.

1. What is a YAML file, and how does Ansible use it?

* YAML full form is Yet Another Markup Language, Ansible uses YAML syntax for coding the Ansible Playbooks.
* It is English like language and is easy to write, read, and understand than other language formats such as JSON and XML.
* 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.
* Instead of writing complex scripts, Ansible users create high-level plays to ensure a host or group of hosts are in a particular state. A play performs a series of tasks on the hosts, in the order specified by the play.
* These plays are expressed in YAML format in a text file.

1. What are the different types of Ansible tasks?

* The task is a unit action of Ansible. we can say task are small small unit of ansible action. so ansible is going to perform some action and that action can be written in form of task.
* we can execute a single task once with an ad hoc command.
* It helps by breaking a configuration policy into smaller files or blocks of code. These blocks can be used in automating a process.
* you can have multiple small small task and create a playbook. For example, to install a package or update a software
* Install <package\_name>, update <software\_name>
* eg.  
  name: just an example  
  hosts: webservers  
  tasks:  
  - first  
  - second  
  - third

1. What are the best ways to use YAML files in high-level programming languages like Java, Python, and others?

* YAML is supported in most programming languages and can be easily integrated with user programs.
* PyYAML is a Python module that we can use as it provides a range of methods to perform several operations on the YAML file.
* For that we need to install module with command: pip install pyyaml
* We can easily convert the YAML file into the Dictionary and read its content.
* With the help of the YAML module, we can read write complex configuration YAML files, serializing and persisting YAML data.

Create below yaml and then run below python and you can see the output.

|  |
| --- |
| Sample\_poc.yaml  # YAML Document starts with ---  # Comments start with #  UserName: Ronak  Password: ineuron\_poc  phone: 999888777  Skills:  -Python  -SQL  -IBM MQ  -ApaceKafka |

|  |
| --- |
| import yaml    from yaml.loader import FullLoader  #open yaml file in read  with open('sample\_poc.yaml', 'r') as f:  yaml\_data = yaml.load(f, Loader=FullLoader)  print(yaml\_data) |

* While in In JAVA we can use the Jackson module which also parses XML and JSON. For e.g

|  |
| --- |
| //Let's first declare Topic class with few attributes like name, total\_score, user\_score, //sub\_topics  List<Topic> topics = new ArrayList<Topic>();  topics.add(new Topic("Bit Manipulation", 10, 6));  topics.add(new Topic("Masking", 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); |

|  |
| --- |
| // Now in yaml file/playbook we can create as below  ---  name: "DS & Algo"  total\_score: 35  user\_score: 24  sub\_topics:  - name: "Bit Manipulation"  total\_score: 10  user\_score: 6  - name: "Masking"  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); |

1. How to set up a jump host to access servers having no direct access?

* this can be achieved by setting 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=15.0.2.1

staging2 ansible\_host=15.0.2.2

* 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 GATWAY group ansible will append these arguments to the command line.

1. How can I use encrypted files to automate password entry in a playbook?

* Above can be achieved by lets keep a separate script that specifies the passwords. Now we need to print a password to stdout to work without errors. that be done as below eg.

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

* One more way is we can have a password file for all the passwords of encrypted files will be saved and ansible should make a call to fetch those when required. so to do that below is the syntax

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

1. What are Ansible callback plugins?

* Callback plugins is used to control most of the output what we see while running command line programs.
* But it can also be used to add additional output.
* Callback plugins enable adding new behaviors to Ansible when responding to events.
* It can used to add additional output or integrate with other tools and events to a storage backend.
* Ansible comes with a collection of callback plug-ins that you can enable in the ansible.cfg file through the callback\_whitelist directive.
* eg:
* [ronak@ineuron\_poc ~]$ **cat ansible.cfg**  
  [defaults]  
  inventory=inventory  
  remote\_user=devops  
  **callback\_whitelist=timer, profile\_tasks, cgroup\_perf\_recap**

1. What is Ansible Inventory and what are the many types of it?

* The inventory file is the hosts and group of hosts on which the modules, tasks, commands in a playbook operates.
* The default location for the inventory file is /etc/ansible/hosts.
* Managed hosts are listed in an inventory
* 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]
* staging\_1 ansible\_host=101.0.2.1
* staging\_2 ansible\_host=101.0.2.2
* Dynamic inventory is generated by a script written in Python or any other programming language or by using plugins.
* In a cloud setup, static inventory file configuration will fail since IP addresses change once a virtual server is stopped and started again.
* Let us just see one poc done create an inventory as below which has all range and groups as below

[webservers]  
server[a:d].lab.example.com  
[raleigh]  
servera.lab.example.com  
serverb.lab.example.com  
[mountainview]  
serverc.lab.example.com  
[london]  
serverd.lab.example.com  
[development]  
servera.lab.example.com  
[testing]  
serverb.lab.example.com  
[production]  
serverc.lab.example.com  
serverd.lab.example.com  
[us:children]  
raleigh  
mountainview

* Use the ansible all -i inventory --list-hosts command to list all managed hosts.

[ronak@ineuron\_poc deploy-inventory]$ **ansible all -i inventory --list-hosts**

hosts (4):

servera.lab.example.com

serverb.lab.example.com

serverc.lab.example.com

serverd.lab.example.com

* [ronak@ineuron\_poc deploy-inventory]$ **ansible ungrouped -i inventory --list-hosts**

[WARNING]: No hosts matched, nothing to do

hosts (0)

* you much include -i inventory option in ansible command because this makes ansible use your inventory file in the current working directory instead of the system /etc/ansible/hosts inventory file

1. What is an Ansible Vault, exactly?

* Ansible Vault is used to encrypt and decrypt any structured data file used by Ansible. To use Ansible Vault, a command-line tool named ansible-vault is used to create, edit, encrypt, decrypt, and view files.
* Ansible Vault can encrypt any structured data file used by Ansible.
* This can include inventory variables, included variable files in a playbook, variable files passed as arguments when executing the playbook, or variables defined in Ansible roles.

Below is the poc:

* [ronak@ineuron\_poc ~]$ ansible-vault create secret.yml
* New Vault password: redhat
* Confirm New Vault password: redhat
* [ronak@ineuron\_poc ~]$ ansible-vault view secret1.yml

Vault password: **secret**

my\_secret: "yJJvPqhsiusmmPPZdnjndkdnYNDjdj782meUZcw"

* [ronak@ineuron\_poc ~]$ ansible-vault decrypt secret1.yml --output=secret1-decrypted.yml
* Vault password: redhat
* Decryption successful