1.flow oall the tools - CI\CD complete flow

Code in git -> Jenkins Jobs (maven for compiling and Ansible to push the code)

2.For java app what are the best tools which is used for CI and CD tools.-----

Jenkins + Ansible

3.Advantage of puppet ---

Configuration Management

4.High level arch of chef -----

chef workstation -> Chef server -> Chef Node

5.Default name given to the repository (origin is the default name given to any repository) ----

6.Cloud based deployment experience -----

AWS

7.My contribution in the recent project and day to day activity ----

Writing puppet modules

8.Various plugin used in ur project ----

Backup, git , Delivery pipeline, Build pipeline , Cucumber , dashboard, Build graph etc.

9.How do u do roll back deployment, for eg- installed an app with version 6 and for some reason ----

By using git tags.

10.I have to roll back to version 5.9, how to roll back to version 5.9. ----

11.What is the achievement you have done in ur exp of devops ----

12.Automation done using puppet,chef and ansible ----

13.What problem CI solves, advantage of CI. ----

What code is updated , It gets compiled automatically.

14.Different scripts used in ur project and what is the purpose of the scripts ----

15.How u have implemented devops in ur project ----

started writing puppet modules for infrastucture.

Ansible

16.Y do u use ansible for automation deployment, jenkins can also do the auto deployment, then y ansible is used y not jenkins. - Jenkins is a CI tool and ansible is very easy for pushing code to multiple systems.

17.different types of inventories in ansible - /etc/ansible/hosts, group_vars and host_vars.

18.Diff bet ansible and chef - ansible is agent less ans works on push method and chef works on pull method and has server-client.

19.in which language we write playbooks in ansible -

YAML

20.if we want system information of machine how we will get the data with ansible like we use facter in puppet - using module setup.

21.Activity that we use with ansibles or advantage Functionality or purpose of ansible using in your project - idempotency , simple because of modules.

22.Ansible Configuration files and process required to take control of other 3 machines - controller and entried in /etc/ansible/hosts.

23.Functionality use with ansible in ur project and what is the adv of having ansible in ur project---

For deployments and provisioning resources on AWS.

24.Activity we can do using ansible like main advantage of using ansible --- idempotency.

25.How to launch an LDAP server using ansible ---

Used LDAP module from galaxy as a reference and modified it according to our

environment.

26.How ansible works.. and what are the playbook written other than the basic playbooks --- we are using even to provision AWS resources.

27.For example there are 4 aps server running and 4 websers running and we have a new code change and this change should automatically copy to these different servers using ansible, how I will come to know new code is generated. --- use Jenkins with POLL SCM option and after artifact is generated using ansible to push suing copy module. (make sure serial: 1).

28.Have u done any automation like app goes down then for self healing of the app kind of the thing. Have u written any playbook. (make sure serial: 1). We have written for Tomcat.

Maven and Ant

29.what is maven from where we get the pom.xml for the jobs.—

Maven is a build tool.

```
mvn archetype:generate
```

30.what is pom.xml ---

Any Maven project must have a `pom.xml` file. POM is the Maven project descriptor just like the `web.xml` file in your Java EE web application, or the `build.xml` file in your Ant project.

31.maven and ant difference and what are the different scenario where it is used----

Difference between Ant and Maven. Ant and Maven both are build tools provided by Apache. The main purpose of these technologies is to ease the build process of a project. Ant doesn't has formal conventions, so we need to provide information of the project structure in build.xml file.

32.what is ant and from where we get the build.xml for the jobs.---

33.how pom.xml will read the variables which we use in the jenkin like version.----

By invoking maven

34.what is dependency in pom.xml---

POM inheritance

35.Invoke maven task is there in post build option.

Jenkins

36.are we in admin part of jenkins like installation of jenkins ---- yes

37different ways to install a plugin's in jenkins - using manage plugins.----

38.how you created jobs in jenkins other than the GUI like new item option.. other option is job dsl. ---- using Jenkins API

39.Why jenkin is required if we can do the same thing with script like automatic. - Continous Integration and its plugin feature.

40.How many jobs are there in jenkins (for eg – ant , maven) - ant is default Maven you have to install, , Gradle.----

41.steps for the installation of jenkins tomcat server (for tomcat server install command is different) - see our copy.yml.

42.yum install Jenkins will run on which server ? - Master server.

43.how do you do a deployemnt. - using shell scripts

44.are you doing the deployment in an application server - yes.

45.What is sonar qube and How to Integrate sonar qube with jenkins - code metrics.

46.Main reason of using jenkins - CI.

47.Commands to start jenkin manually through command prompts like how to up the server - service jenkins start.

48.How to Safe restart the jenkins - Restart when no jobs are running.

49.How to create a backup in jenkins, what is the plug in name complete process. From which directory it will take the back up (jenkin_home) directory. - Backup Plugin.

50.Step to setup a jenkins and how to create a new job - New item-> Frees style.

51.How to deploy a Custom build in jenkins - usinfg build tools such as ant and maven.

52.Where we store the artifacts which gets generated from jenkins - local and nexus .

53.How to bypass the github like directly using the code and put it in jenkins--- t

54.Jenkins used a entriprised or a open source u used - ---open source.

55.Different Pluging used in jenkins ---build pipeline, Backup, git .

Git

56.pull request in git hub - ---git pull

57.fork a repository in git hub----

Forking Projects

If you want to contribute to an existing project to which you don’t have push access, GitHub
encourages forking the project. When you land on a project page that looks interesting and
you want to hack on it a bit, you can click the “fork” button in the project header to have
GitHub copy that project to your user so you can push to it.
This way, projects don’t have to worry about adding users as collaborators to give them
push access. People can fork a project and push to it, and the main project maintainer can pull
in those changes by adding them as remotes and merging in their work.

58.Diff between svn and git ---svn is centralized version control and git is Distributed Version Control.

Linux

59.How to delete different lines in a file in linux eg:- line 1,4,6,10. ---

Sed ‘1d;4d;6d;10d’ filename

60.difference between soft link and hard link in linux---

Hard Link acts like a mirror copy of the original file. These links share the same inodes. Changes made to the original or hard linked file will reflect in the other. When you delete Hard Link nothing will happen to the other file. Hard links can't cross file systems.

Soft Link is actual link to the original file. These Links will have a different Inodes value. Soft link points to the original file so if original file is deleted then the soft link fails. If you delete the Soft Link, nothing will happen to file. The reason for this is, the actual file or directory’s inode is different from the "soft link" created file's inodes. Soft links can cross file systems

61.can we create a soft link and hard link for file and dir---

Hard linkink of directorties is not possible

62.need to create a user and dont want the user to get the shell access--- put shell as /bin/false

63.Top command in linux--- top (to check memory, process usage)

63.How to check dynamic log files in linux--- tail –f filename

Shell

64.Define a shell script---

A shell script is a text file that contains a sequence of commands for a UNIX-based operating system. It's called a shell script because it combines into a "script" in a single file a sequence of commands that would otherwise have to be presented to the system from a keyboard one at a time.

65.What are the shell script written in ur projects---

66.How to execute a shell file in debug mode---

Set +x

67.File contains my name 10 times we have to write the script where I have to count the number of times my name appears in the file. ---

vi abc.txt

#! /bin/bash

grep $1 | wc –l

# sh abc.sh filename

Nagios

68.What types of monitoring and what are the monitoring u have done on daily basis - Responding to the alerts. Adding new nodes and services on those nodes. ---

I have written puppet nrpe module.

69.How nagios works or how u monitor the servers - ---NRPE.

70.written any pliugin in nagios ---- **I have plugins genereated from Nagios plugin installation and plugins from exchange.nagios.com.**

Puppet:

71.What is puppet?---

Puppet is a configuration management tool though it can be used fr orchestration also.

72.How to install puppetserver?---

Puppet is available in both open source and Enterprise edition.
Open source is available in EPEL.
Enterprise has to be downloaded for puppetlabs site.

73.How to setup enviornment in puppet?----

By setting Modulepath and Manifest destinations in puppet.conf.

74.What is puppet.conf/ what is puppet configuration file name? ---
Puppet.conf is main configuration file in Puppet.

75.Sections in puppet.conf file? ---
main, master and agent

76.How to setup agent in puppet? ----
By installing puppet rpm.

77.Where do we mention puppet server information in puppet agent? ---
In puppet.conf under agent section.

78.What port puppetmaster listens? ---
see below notes

79.What port puppet agent listens? ---

see below notes

80.Where do we set puppet agent node definition in puppetmaster? ----
in site.pp file.

81.Default & enviornment both? ---
production

82.What are puppet manifests? ---

file should have and extension of .pp file and inside this file we will define resources using puppet DSL.

83.Location of puppet manifests, default and environment both? ---
Are mentioned in puppet.conf

84.Name 10 puppet resources? ----
check your cheat sheet.

85.What are selectors? ---

condition in puppet can be set using if/else, case and selectors.

86.What are classes in puppet and how to define it? ---

class is a collection of resources. is specified inside manifest file.

87.Installing multiple packages in by using variables? ---
variables can be defined using $variablename.

ex: $name1 = 'httpd'

```
class httpd {
        package {'httpd':
            package_name => $name1,
                     }
}
```

89.What modules in puppet and how to create it? ----

Modules is collection of the files, templates, manifests required to implement a scenario.

Example: Setting up Web server

90.How to use puppetforge modules? ---

To download puppet modules using puppet module install.

91.Conditional statements in puppet? ---

if/else, case and selectors/

92.How to finetune or customize puppetforge modules? ---

Download the module from puppetlabs and modify it according to our environment.

Chef :

93.what is chef work flow----

Workstation -> Server -> Nodes

94.how we get the properites of node----

Ohai

Tool used to detect a_ributes on a node and then provide attributes to chef-client at the start of every chef-client run

95.what is resourse ----

Resources:
A statement of configura>on policy within a recipe

Describes the desired state of an element in the infrastructure and steps needed to configure

96.what is diff b/w chef server and chef client----

Chef--Client:
Agent that runs locally on the node that is registered with the chef server

Chef Server:
Chef server is the hub for all configura>on data, stores cookbooks, and the policies applied to the node

97.what is run list-----

Nodes receive their policy based off of roles and individual node configura>ons
A run list defines the order in which you want your recipes to run during convergence

98.what is diff b/w cookbook and recipe----

Recipes are made up of a collection of resources
Cookbooks are made up of a collection of recipes

99.how bootstrap will work & what is process----

Nodes should be bootstrapped and managed from the workstation

Nodes should be assigned roles and environments

A_ributes specific to roles/environments should be configured accordingly

100.which language chef deploy?

ruby

What is Puppet?

Infrastructure automation and configuration management tool

Enforces the defined state of the infrastructure

Puppet can automate tasks on thousands of machines.

Puppet enables infrastructure as code

Puppet allows configuration consistency across nodes

Puppet enables quick provisioning of new machines in an environment

Puppet allows DevOps admins to write declarative instrucions using the Puppet language

DevOps admins write code using the Puppet DSL to express the desired state of a node

Code is written inside of classes and classes are assigned to node

Node classifica0on is the process of assigning a class to a node for processing

Puppet Ports

- 3000 - Web-based installer of puppet master
- 8140 puppet requests
- 61613 - orchestration requests
- 443 for console managemet

Example:
Puppet language DSL

```
{
        Class motd
                {
                        file { "/etc/motd":
                                ensure => 'file',
                                source =>
                        "puppet:///modules/motd/motd",
                }
}
```

Puppet Master

Compile and serve configura0on catalogs to agent nodes

Issue orchestration commands via MCollec0ve

Availability of puppet enterprise web interface console

Collect data and compile reports from agent nodes

What is a catalog?
The catalog is a document downloaded from the puppet master to the agent that describes the desired state for each resource that should be managed. It may also specify dependency information for the resources that should be managed in a certain order. This is essentially a compiled version of the DSL and is compiled on the Puppet master and stored in PuppetDB if PuppetDB is used.

Puppet enterprise includes MCollective which is an orchestration system that allows the admin to invoke different commands in parallel across a select group of nodes.

Puppet Process

Install Puppet master and Puppet agents
Create configurations (modules/classes) that declare a resources desired state
Assign configurations to nodes

Puppet nodes (agent)
Nodes are any virtual or physical system that is able to run Puppet Agent and is specifically supported by puppet agent.

Required Ports:

8140 for puppet requests

61613 for orchestra0on requests

Puppet.conf

Puppet Enterprise: /etc/puppetlabs/puppet/puppet.conf

Puppet OpenSource /etc/puppet/puppet.conf

Config Sections:

[main] – Global section used by all services/servers

[master] – Use by Puppet master and puppet cert command

[agent] – Used by the Puppet agent service (even if the agent runs on the master)

[user] – Used most commonly by the Puppet apply command

Resource Abstraction Layer

A system configuration is a collection of resources that make up the state of your system

Users

Cronjobs

Packages installed

Services

Etc.

Resources

When building modules we are using the puppet DSL to declare the desired state of resources on a node .Fundamentally all we are doing with Puppet is managing resources on a large and automated scale.

In Puppet we are using resource types to define instances of a resource on a node

Chef

Common Chef Terminology

Recipes:
Fundamental configura>on element within an organiza>on

Cookbook:
Defines a scenario and is the fundamental unit of configura>on and policy distribu>on
Chef--Client:
Agent that runs locally on the node that is registered with the chef server

Convergence:
Occurs when chef--client configures the system/node based off the information collected from chef--
server

Configuration Drift:
Occurs when the node state does not reflect the updated state of polices/configura>ons on the chef server

Resources:
A statement of configura>on policy within a recipe Describes the desired state of an element in the infrastructure and steps needed to configure

Provider:
Defines the steps that are needed to bring the piece of the system from its current state to the desired state

Attributes:
Specific details about the node, used by chef--client to understand current state of the node, the state of the node on the previous chef--client run, and the state of the node at the end of the client run
Data--bags:
A global variables stored as JSON data and is accessible from the Chef server

Workstation:
A computer configured with Knife and used to synchronize with chef--repo and interact with chef server
Chef Server:
Chef server is the hub for all configura>on data, stores cookbooks, and the policies applied to the node

Knife:
Command line tool which provides an interface between the local chef--repo and chef--server
client.rb

Configuration file for chef--client located at /etc/chef/client.rb on each node

Ohai:

Tool used to detect a_ributes on a node and then provide a_ributes to chef--client at the start of every chef--client run
Node Object:

Consists of run--list and node attributes that describe states of the node

Chef--Repo:

Located on the workstation and installed with the starter kit, should be synchronized with a version control system and stores Cookbooks, roles, data bags, environments, and configuration files

Organization:

Used in chef enterprise server to restrict access to objects, nodes environments, roles, data--bags etc

Environments:

Used to organize environments (Prod/Staging/Dev/QA) generally used with cookbook versions

Idempotence

Means a recipe can run multiple times on the same system and the results will always be identical