

GIT:-

1) What is GIT?

A) GIT is a distributed version control system and source code management (SCM) system with an importance to handle small and large projects with speed and efficiency.

2) What is a repository in GIT?

A) A repository contains a directory named .git, where git keeps all of its metadata for the repository. The content of the .git directory are private to git.

3) What is the command you can use to write a commit message?

A) The command that is used to write a commit message is “git commit -m”. You can use “git add<file>” before git commit -m if new files need to be committed for the first time.

4) What is the difference between GIT and SVN?

A) The difference between GIT and SVN is

a) Git is called as distributed version control system where as SVN is called as centralized version control system.

b) GIT does not support ‘commits’ across multiple branches or tags. Subversion allows the creation of folders at any location in the repository layout.

c) For Git network is required at the time of push & pull only where as for SVN network is required all time.

5) What are the advantages of using GIT?

a) Data redundancy and replication

b) High availability

c) Only one git directory per repository

d) Superior disk utilization and network performance

e) Collaboration friendly

f) Any sort of projects can use GIT

6) What language is used in GIT?

A) GIT is fast, and ‘C’ language makes this possible by reducing the overhead of runtimes associated with higher languages.

7) What is the function of ‘GIT PUSH’ in GIT?

A) ‘GIT PUSH’ updates contents to remote repository from local repository.

8) Why GIT better than Subversion?

A) GIT is an open source version control system; it will allow you to run 'versions' of a project, which show the changes that were made to the code overtime also it allows you keep the backtrack if necessary and undo those changes. Multiple developers can checkout, and upload changes and each change can then be attributed to a specific developer.

9) What is "Staging Area" or "Index" in GIT?

A) Before completing the commits, it can be formatted and reviewed in an intermediate area known as 'Staging Area' or 'Index'.

10) What is GIT stash?

A) GIT stash is a temporary memory for storing data from staging/index area on the stack for later and gives you back a clean working directory. So in case if you are in the middle of something and need to jump over to the other job, and at the same time you don't want to lose your current edits then you can use GIT stash.

11) What is GIT stash drop?

A) When you are done with the stashed item or want to remove it from the list, run the git 'stash drop' command. It will remove the last added stash item by default, and it can also remove a specific item if you include as an argument.

12) What is the function of git clone?

A) The git clone command creates a copy of an existing Git repository. To get the copy of a central repository, 'cloning' is the most common way used by programmers.

13) What is the function of 'git config'?

A) The 'git config' command is a convenient way to set configuration options for your Git installation. Behaviour of a repository, user info, preferences etc. can be defined through this command.

14) What does commit object contain?

- A)
- a) A set of files, representing the state of a project at a given point of time
 - b) Reference to parent commit objects
 - c) An SHAI name, a 40 character string that uniquely identifies the commit object.

15) How can you create a repository in Git?

A) In Git, to create a repository, create a directory for the project if it does not exist, and then run command "git init". By running this command .git directory will be created in the project directory, the directory does not need to be empty.

16) What is 'head' in git and how many heads can be created in a repository?

A) A 'head' is simply a reference to a commit object. In every repository, there is a default head referred as "Master". A repository can contain any number of heads.

17) What is the purpose of branching in GIT?

A) The purpose of branching in Git is that you can create your own branch and jump between those branches. It will allow you to go to your previous work keeping your recent work intact.

18) What is the common branching pattern in GIT?

A) The common way of creating branch in GIT is to maintain one as "Main" branch and create another branch to implement new features. This pattern is particularly useful when there are multiple developers working on a single project.

19) How can you bring a new feature in the main branch?

A) To bring a new feature in the main branch, you can use a command "git merge" or "git pull command".

20) What is a 'conflict' in git?

A) A 'conflict' arises when the commit that has to be merged has some change in one place, and the current commit also has a change at the same place. Git will not be able to predict which change should take first.

21) How can conflict in git resolved?

A) To resolve the conflict in git, edit the files to fix the conflicting changes and then add the resolved files by running "git add" after that to commit the repaired merge, run "git commit". Git remembers that you are in the middle of a merger, so it sets the parents of the commit correctly.

22) To delete a branch what is the command that is used?

A) Once your development branch is merged into the main branch, you don't need development branch. To delete a branch use, the command "git branch -d branchname".

23) What is another option for merging in git?

A) "Rebasing" is an alternative to merging in git.

24) What is the syntax for "Rebasing" in Git?

A) The syntax used for rebase is "git rebase [new-commit]" "

25) What is the difference between 'git remote' and 'git clone'?

A) 'git remote add' just creates an entry in your git config that specifies a name for a particular URL. While, 'git clone' creates a new git repository by copying and existing one located at the URL.

26) What is GIT version control?

A) With the help of GIT version control, you can track the history of a collection of files and includes the functionality to revert the collection of files to another version. Each version captures a snapshot of the file system at a certain point of time. A collection of files and their complete history are stored in a repository.

27) What is Subgit? Why to use Subgit?

A) 'Subgit' is a tool for a smooth, stress-free SVN to Git migration. Subgit is a solution for a company -wide migration from SVN to Git that is:

- a) It is much better than git-svn
- b) No requirement to change the infrastructure that is already placed
- c) Allows to use all git and all sub-version features
- d) Provides genuine stress –free migration experience.

28) What is the function of 'git diff ' in git?

A) 'git diff ' shows the changes between commits, commit and working tree etc.

29) What is 'git status' is used for?

A) As 'Git Status' shows you the difference between the working directory and the index, it is helpful in understanding git very easily.

30) What is the difference between the 'git diff 'and 'git status'?

A) 'git diff' is similar to 'git status', but it shows the differences between various commits and also between the working directory and index.

31) What is the function of 'git checkout' in git?

A) A 'git checkout' command is used to go from one branch to another branch.

32) What is the function of 'git rm'?

A) To remove the file from the staging area and also off your disk 'git rm' is used.

33) What is the function of 'git stash apply'?

A) When you want to continue working where you have left your work, 'git stash apply' command is used to bring back the saved changes onto the working directory.

34) What is the use of 'git log'?

A) To find specific commits in your project history- by author, date, content or history 'git log' is used.

35) What is 'git add' is used for?

A) 'git add' adds file changes in your existing directory to your index.

36) What is the function of 'git reset'?

A) The function of 'Git Reset' is to reset your index as well as the working directory to the state of your last commit.

37) What does 'hooks' consist of in git?

A) This directory consists of Shell scripts which are activated after running the corresponding Git commands. For example, git will try to execute the post-commit script after you run a commit.

38) Explain what is commit message?

A) Commit message is a feature of git which appears when you commit a change. Git provides you a text editor where you can enter the modifications made in commits.

39) How can you fix a broken commit?

A) To fix any broken commit, you will use the command "git commit --amend". By running this command, you can fix the broken commit message in the editor.

40) What is 'bare repository' in GIT?

A) To co-ordinate with the distributed development and developers team, especially when you are working on a project from multiple computers 'Bare Repository' is used. A bare repository comprises of a version history of your code.

41) Name a few Git repository hosting services

A)

- BitBucket
- GitHub
- GitEnterprise
- SourceForge.net
- Visual Studio Online

Maven Interview Questions

1. What is Maven?

A) Maven is a project management and build tool. Maven provides developers a complete build life cycle framework.

2. What are the aspects Maven manages?

A) Maven provides developers ways to manage following –

- Builds
- Documentation
- Reporting
- Dependencies
- SCMs
- Releases
- Distribution
- mailing list

3. How does one understand the version of mvn you're using?

A) mvn -version

4. What is POM?

A) POM stands for Project Object Model. It's fundamental Unit of Work in Maven. It is an XML file. It's continuously resides within the base directory of the project as xml. It contains data regarding the project and numerous configuration details to create the project(s).

5. What information does POM contain?

A) POM contains the a number of the subsequent configuration info –

- project dependencies
- plugins
- goals
- build profiles
- project version
- developers
- mailing list

6. What is Maven artifact?

A) An artifact may be a file, sometimes a JAR that gets deployed to a expert repository. A expert build produces one or a lot of artifacts, like a compiled JAR and a “sources” JAR.

Each artifact has a group ID (usually a reversed domain name, like com.example.foo), an artifact ID (just a name), and a version string. The three together uniquely identify the artifact. A project's dependencies are specified as artifacts.

7. What is Maven Build Life cycle?

A) A Build Life cycle could be a well outlined sequence of phases that outline the order during which the goals are to be dead. Here phase represents a stage in life cycle.

8. Name the 3 build life cycle of Maven.

A) The three build life cycles are –

clean: cleans up artifacts created by prior builds.

default (or build): This is employed to create the applying.

site: generates site documentation for the project.

9. What is the command to quickly build your Maven site?

A) mvn site

10. What would the command mvn clean do ?

A) This is command removes the target directory with all the build data before starting the build process.

11. What are the phases of a Maven Build Life cycle?

A) Following are the phases –

- Validate: The project is correct & every one necessary info is accessible.
- Compile: The ASCII text file of the project.
- Test: Check the compiled ASCII text file employing an appropriate unit checking framework.
- Package: Take to the compiled code & package it in its distributable format like a JAR.
- Integration-test: To perform initegration tests if required
- Verify: To run any checks to verify the package is valid and meets quality criteria.
- Install: The package into the native repository, to be used as a dependency in different comes domestically.
- Deploy: Worn out associate degree integration , copies the ultimate package to the remote repository for sharing with different developers and comes.

12. What is a goal in Maven terminology?

A) A goal represents a particular task that contributes to the building and managing of a project.

13. What would this command do mvn clean dependency:copy-dependencies package?

A) This command will clean the project, copy the dependencies and package the project (executing all phases up to package).

14. What phases does a Clean Life cycle consist?

A) The clean life cycle consists of the following phases –

- Pre-clean
- Clean
- Post-clean

15. What phases does a Site Life cycle consist?

A) The phases in Site Life cycle are –

- Pre-site
- Site
- Post-site
- Site-deploy

16. What is Build Profile?

A) A Build profile could be a set of configuration values. Using a build profile, you'll be able to customize build for various environments like Production v/s Development environments.

17. What are different types of Build Profiles?

A) Build profiles are of three types –

Per Project – Defined in the project POM file and pom.xml.

Per User–Defined in Maven settings.xml file (%USER_HOME%/.m2/settings.xml)

Global – Defined in Maven global settings xml file (%M2_home%/conf/settings.xml)

18. What is a Maven Repository?

A) A repository is a place i.e. directory wherever all the project jars, library jar, plugins or the other project specific artifacts square measure hold on and might be employed by adept simply.

19. What types of Maven repository?

A) Maven repository square measure of 3 types: native, central, remote

20. What is local repository?

A) Maven native repository could be a folder location on your machine. It gets created after you run any maven command for the primary time. Maven native repository keeps your project's all dependencies (library jars, plugin jars etc).

21. What is the default location for your local repository?

A) ~/m2/repository.

22. What is the command to install JAR file in local repository?

A) mvn install

23. What is Central Repository?

A) repository provided by the Maven community. the contains in a large number of commonly used libraries. When Maven does not find any dependency in a local repository, it starts searching in a central repository using the following URL: <http://repo1.maven.org/maven2/>.

24. What is a Remote Repository?

A) Sometimes, genius doesn't notice a mentioned dependency in a very central repository furthermore then it stops the build method and output error message to console. To prevent such state of affairs, genius provides an inspiration of Remote Repository that is that the developer's own custom repository containing needed libraries or alternative project jars.

Jenkins Interview Questions

Q1. What is the difference between Jenkins and Bamboo?

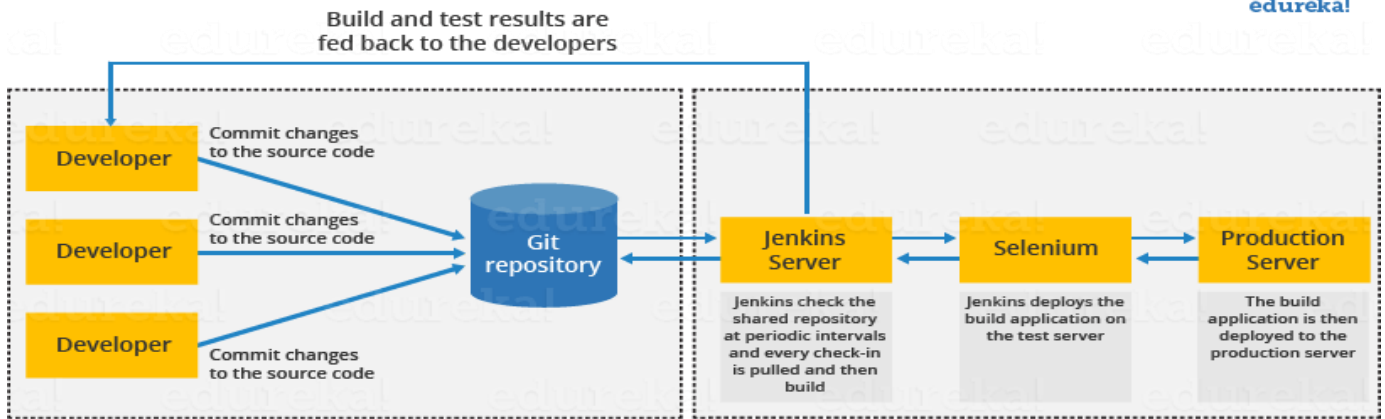
Parameters	Jenkins	Bamboo
Open Source	Jenkin is open-source	Bamboo is not open source
Pricing Logic	Jenkin is completely free	It charges for the number of build agents required
Operating System	Windows, Ubuntu, Red Hat, Mac OS	Windows, Linux, Solaris
Browsers	Chrome, Firefox, Internet Explorer	Firefox, Chrome, Safari, Edge
Plugin Support	Yes, It supports a lot of plugins	It does not support many plugins as compared to Jenkins
Support	Being open-source, it has a lot of support from communities	It has less support as compared to Jenkins

Q2. What is Jenkins?

A) Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies.

Q3. Define the process of Jenkins?

- First, a developer commits the code to the source code repository. Meanwhile, the Jenkins server checks the repository at regular intervals for changes.
- Soon after a commit occurs, the Jenkins server detects the changes that have occurred in the source code repository. Jenkins will pull those changes and will start preparing a new build.
- If the build fails, then the concerned team will be notified.
- If the build is successful, then Jenkins deploys the build in the test server.
- After testing, Jenkins generates feedback and then notifies the developers about the build and test results.
- It will continue to check the source code repository for changes made in the source code and the whole process keeps on repeating.



Q4. What are the benefits of using Jenkins?

- I will suggest you include the following benefits of Jenkins if you can recall any other benefit apart from the below-mentioned points you can include that as well.
- At the integration stage, you can cache build failures.
- For each change in the source code, you generate an automatic build report notification.
- To notify developers about build report success or failure, Jenkins integrates with the LDAP mail server.
- Achieves continuous integration agile development and test driven development.
- With simple steps, you can automate the maven release project.
- Easy tracking of bugs at an early stage in a development environment than production.

Q5. What are the pre-requisites for using Jenkins?

A) The answer to this is pretty straightforward. To use Jenkins you require:

- A source code repository which is accessible, for instance, a Git repository.
- A working build script, e.g., a Maven script, checked into the repository.

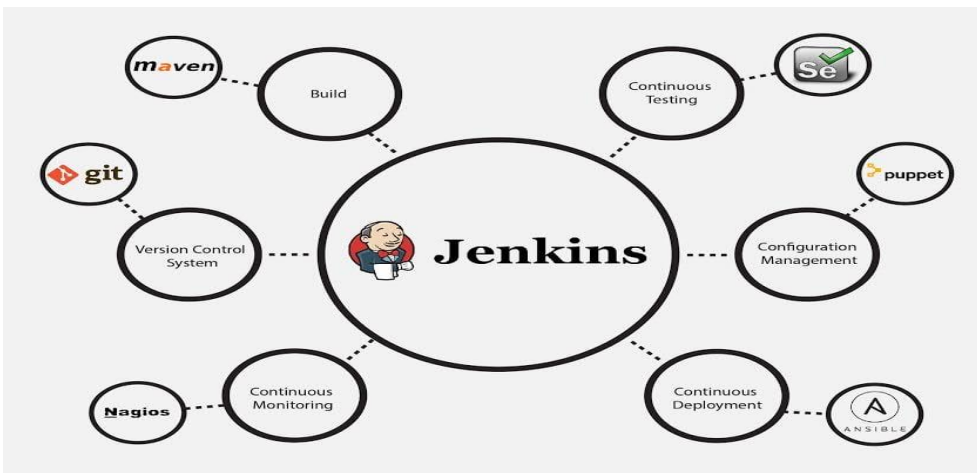
Q6. What is the relation between Hudson and Jenkins?

A) You can just say Hudson was the earlier name and version of current Jenkins. After some issues, they renamed the project from Hudson to Jenkins.

Q7. Mention some of the useful plugins in Jenkins

A) Below I have mentioned some important Plugins:

- Maven 2 project
- Git
- Amazon EC2
- HTML publisher
- Copy artifact
- Join
- Green Balls



These Plugins I feel are the most useful plugins, if you want to include any other Plugin that is not mentioned above, you can add that as well, but make sure you first mention the above-stated plugins and then add your own.

Q8. How do you install Jenkins?

To install Jenkins, you just need to follow these five steps:

1. **Install Java Version 8** – Jenkins is a Java based application, hence Java is a must.
2. **Install Apache Tomcat Version 9** – Tomcat is essential to deploy Jenkins war file.
3. **Download Jenkins war File** – This war is must to install Jenkins.
4. **Deploy Jenkins war File** – You deploy Jenkins war file using Tomcat to run Jenkins.
5. **Install Suggested Plugins** – Install a list of plugins suggested by Jenkins.

Once the installation is complete, you will be able to see the Jenkins dashboard.

Q9. What are the two components that you can integrate Jenkins with?

A) According to me, the integration of Jenkins is possible with the following:

- Version Control system like GIT, SVN.
- Build tools like Apache Maven.

Q10. What is Maven? What is the benefit of integrating Maven with Jenkins?

A)

[Maven](#) is a build management tool. It uses a simple pom.xml to configure all the dependencies needed to build, test and run the code. Maven manages the full lifecycle of a test project. Once integrated with Jenkins, the maven Webdriver will build the project and execute all tests efficiently.

Q11. Which SCM tools Jenkins supports?

Here are some of the Source Code Management tools supported by Jenkins:

- Git
- Subversion
- CVS
- Mercurial
- Perforce

Q12. How will you define Post in Jenkins?

Post is a section that contains several additional steps that might execute after the completion of the pipeline. The execution of all the steps within the condition block depends upon the completion status of the pipeline. The condition block includes the following conditions – **changed success, always, failure, unstable and aborted**.

Q13. What are Parameters in Jenkins?

Parameters are supported by Agent section and they are used to support various use-cases pipelines. Parameters are defined at the top-level of the pipeline or inside an individual stage directive.

Q14. What is Groovy?

Groovy from Apache is a language designed for the Java platform. It is the native scripting language for Jenkins. Groovy-based plugins enhance Jenkins with great interfaces and build reports that are of dynamic and consistent nature.

Q15. How Can You Clone A Git Repository Via Jenkins?

If you want to clone a Git repository via Jenkins, you have to enter the e-mail and user name for your Jenkins system. Switch into your job directory and execute the “git config” command for that.

Q16. Explain how you can set up Jenkins job.

A)Go to Jenkins top page, select “New Job”, then choose “Build a free-style software project”.

The elements of this freestyle job:

- Optional SCM, such as CVS or Subversion where your source code resides.
- Optional triggers to control when Jenkins will perform builds.
- Some sort of build script that performs the build (ant, maven, shell script, batch file, etc.) where the real work happens.
- Optional steps to collect information out of the build, such as archiving the artifacts and/or recording javadoc and test results.
- Optional steps to notify other people/systems with the build result, such as sending e-mails, IMs, updating issue tracker, etc.

Q17. How to create a backup and copy files in Jenkins?

A)To create a backup all you need to do is to periodically back up your JENKINS_HOME directory. This contains all of your build jobs configurations, your slave node configurations, and your build history. To create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

Q18. How will you secure Jenkins?

- Make sure that the global security is on.
- Check if Jenkins is integrated with my company’s user directory with an appropriate plugin.
- Ensure that the matrix/Project matrix is enabled to fine-tune access.

- Automate the process of setting rights/privileges in Jenkins with custom version controlled script.
- Limit physical access to Jenkins data/folders.
- Periodically run security audits on the same.

Q19. Explain how you can deploy a custom build of a core plugin?

Below are the steps to deploy a custom build of a core plugin:

- Stop Jenkins.
- Copy the custom HPI to **\$Jenkins_Home/plugins**.
- Delete the previously expanded plugin directory.
- Make an empty file called **<plugin>.hpi.pinned**.
- Start Jenkins.

Q20. What you do when you see a broken build for your project in Jenkins?

A)I will open the console output for the broken build and try to see if any file changes were missed. If I am unable to find the issue that way, then I will clean and update my local workspace to replicate the problem on my local and try to solve it.

Q21. What are the various ways in which build can be scheduled in Jenkins?

A)You can schedule a build in Jenkins in the following ways:

- By source code management commits
- After completion of other builds
- Can be scheduled to run at a specified time (crons)
- Manual Build Requests

Q22. What is the use of Pipelines in Jenkins?

A)Pipeline plugin is used in Jenkins for making the Jenkins Pipeline, which gives us the view of stages or tasks to perform one after the other in the pipeline form. It models a series of related tasks. Pipelines help the teams to review, edit and iterate upon the tasks. Pipelines are durable and it can optionally stop and wait for human approval as well to start the next task. A pipeline is extensible and can perform work in parallel. It supports complex CD requirements.

Q23. Explain the terms Agent, post-section, Jenkinsfile

Agent: It is directive to tell Jenkins to execute the pipeline in a particular manner and order.

Post-section: If we have to add some notification and to perform other tasks at the end of a pipeline, post-section will definitely run at the end of every pipeline's execution.

Jenkinsfile: The text file where all the definitions of pipelines are defined is called Jenkinsfile. It is being checked in the source control repository.

Q24. Do you know about cloud computing? How can Jenkins fit into a cloud computing environment? Explain with an example.

Let us take the example of AWS cloud service. Cloud computing services use the CI/CD model so that they can push their work to the customers and constantly receive feedback. Jenkins is

used to automating the CI/CD pipelines. For example, a lot of Jenkins plugins are available for many of the AWS services like Amazon EC2 and ECS.

Q25. What is Kubernetes? How can you integrate Jenkins with Kubernetes?

[Kubernetes](#) is a container orchestration tool. With Kubernetes, one can create multiple container instances to achieve more fault tolerance. You can use the Kubernetes deploy plugin to use it with Jenkins for continuous deploy.

Q26. Have you run automated tests on Jenkins? How is it done?

Yes, this can be done easily. Automated tests can be run through tools like Selenium or maven. Developers can schedule the test runs. Jenkins displays the test results and sends a report to the developers.

Q27. Let us say, you have a pipeline. The first job was successful, but the second failed. What should you do next?

You just need to restart the pipeline from the point where it failed by doing 'restart from stage'.

Q28. What is the use of JENKINS_HOME directory?

All the settings, logs and configurations are stored in the JENKINS_HOME directory.

Q29. What is a backup plugin? Why is it used?

This is a helpful plugin that backs up all the critical settings and configurations to be used in the future. This is useful in cases when there is a failure so that we don't lose the settings.

Q30. What is a trigger? Give an example of how the repository is polled when a new commit is detected.

Triggers are used to define when and how pipelines should be executed.

When Jenkins is integrated with an SCM tool, for example, Git, the repository can be polled every time there is a commit.

- The Git plugin should be first installed and set up.
- After this, you can build a trigger that specifies when a new build should be started. For example, you can create a job that polls the repository and triggers a build when a change is committed.

Q31. How do you define parameters for a build in Jenkins?

A build can take several input parameters to execute. For example, if you have multiple test suites, but you want to run only one. You can set a parameter so that you are able to decide which one should be run. To have parameters in a job, you need to specify the same while defining the parameter. The parameter can be anything like a string, a file or a custom.

Q32. What are the ways to configure Jenkins node agent to communicate with Jenkins master?

There are 2 ways to start the node agent –

- **Browser** – if Jenkins node agent is launched from a browser, a JNLP (Java Web Start) file is downloaded. This file launches a new process on the client machine to run jobs.

- **Command-line** – to start the node agent using the command line, the client needs the executable agent.jar file. When this file is run, it simply launches a process on the client to communicate with the Jenkins master to run build jobs.

Q33. How does Jenkins authenticate users?

There are 3 ways –

- The default way is to store user data and credentials in an internal database.
- Configure Jenkins to use the authentication mechanism defined by the application server on which it is deployed.
- Configure Jenkins to authenticate against LDAP server.

Q34. How can you use a third-party tool in Jenkins?

Below are the steps used for working with a third-party tool in Jenkins.

- First install the third-party software
- Download the plug-in that supports the third-party tool.
- Configure the third-party tool in the admin console.
- Then use the required plug-in from the Jenkins build job.

For different third-party tools, the procedure may vary slightly, because of the difference in configuration settings.

Q35. What are the types of pipelines in Jenkins?

There are 3 types –

1. CI CD pipeline (Continuous Integration Continuous Delivery)
2. Scripted pipeline
3. Declarative pipeline

Q36. What syntax does Jenkins use to schedule build job or SVN polling?

The cron syntax.

Cron syntax is represented using five asterisks each separated by a space. The syntax is as follows – [minutes] [hours] [day of the month] [month] [day of the week]. Example, if you want to set up a cron for every Monday at 11.59 pm, it would be 59 11 * * 1

Q37. What is DevOps and in which stage does Jenkins fit in?

DevOps is a software development practice that blends software development (Dev) with the IT operations (Ops) making the whole development lifecycle simpler and shorter by constantly delivering builds, fixes, updates, and features. Jenkins plays a crucial role because it helps in this integration by automating the build, test and deployment process.

Q38. Do you know any other Continuous Integration tools? How is Jenkins better than any of those?

There are many other CI tools, and the most prominent ones are –

- TeamCity

- Bamboo
- Perforce
- Circle CI
- Go
- Integrity
- Travis CI

There are many more. We cannot say if Jenkins is better than each because each has its own unique features. For example, TeamCity offers great .NET support but is complex and costly, Travis CI is free just like Jenkins and has good documentation too. Bamboo too offers efficient and faster builds but is not completely free and so on.

Q39. Name a Jenkins environment variable you have used in a shell script or batch file.

There are numerous environment variables that are available by default in any Jenkins build job. A few commonly used ones include:

- \$JOB_NAME
- \$NODE_NAME
- \$WORKSPACE
- \$BUILD_URL
- \$JOB_URL

Note that, as new Jenkins plug-ins are configured, more environment variables become available. For example, when the Jenkins Git plug-in is configured, new Jenkins Git environment variables, such as \$GIT_COMMIT and \$GIT_URL, become available to be used in scripts.

Q40. What is Continuous Integration In Jenkins?

In software development, multiple developers or teams work on different segments of the same web application. So in this case, you have to perform integration testing by integrating all modules. In order to do that an automated process for each piece of code is performed on a daily bases so that all your codes get tested. This process is known as continuous integration.

Q41. How do you achieve continuous integration using Jenkins?

Here are the steps –

- All the developers commit their source code changes to the shared Git repository.
- Jenkins server checks the shared Git repository at specified intervals and detected changes are then taken into the build.
- The build results and test results are shared to the respective developers
- The built application is displayed on a test server like Selenium and automated tests are run.
- The clean and tested build is deployed to the production server.

Q42. What is a DSL Jenkins?

The Jenkins “Job DSL / Plugin” is made up of two parts – first, The Domain Specific Language (DSL) itself that allows users to describe jobs using a Groovy-based language, and second, a

Jenkins plugin which manages the scripts and the updating of the Jenkins jobs which are created and maintained as a result.

Q43. How do you create Multibranch Pipeline in Jenkins?

The Multibranch Pipeline project type enables you to implement different Jenkinsfiles for different branches of the same project. In a Multibranch Pipeline project, Jenkins automatically discovers, manages and executes Pipelines for branches that contain a Jenkinsfile in source control.

Q44. What are the types of jobs or projects in Jenkins?

These are the types of jobs/projects in Jenkins –

- Freestyle project
- Maven project
- Pipeline
- Multibranch pipeline
- External Job
- Multi-configuration project
- Github organization

Q45. What is blue ocean in Jenkins?

It is a project that was started with the purpose to rethink the user experience of Jenkins, modeling and presenting the process of software delivery by surfacing information that's important to development teams. This is done with as few clicks as possible, while still staying true to the extensibility that is core to Jenkins. While this project is in the alpha stage of development, the intent is that Jenkins users can install Blue Ocean side-by-side with the Jenkins Classic UI via a plugin

Advanced Questions

Q46. What is Continuous Testing?

Continuous Testing is the process where you execute automated tests as part of the software delivery pipeline. This is done so that you get the feedback on the business risks associated with software as early as possible. It consists of evolving and extending test automation to address the increased complexity and pace of modern application development and delivery.

Continuous Testing means that testing takes place on a continuous basis without any disruption of any kind. In a Continuous DevOps process, a software change is continuously moving from Development to Testing to Deployment. The code undergoes continuous development, delivery, testing and deployment.

Q47. Explain how you can move or copy Jenkins from one server to another?

I will approach this task by copying the jobs directory from the old server to the new one. There are multiple ways to do that, I have mentioned it below:

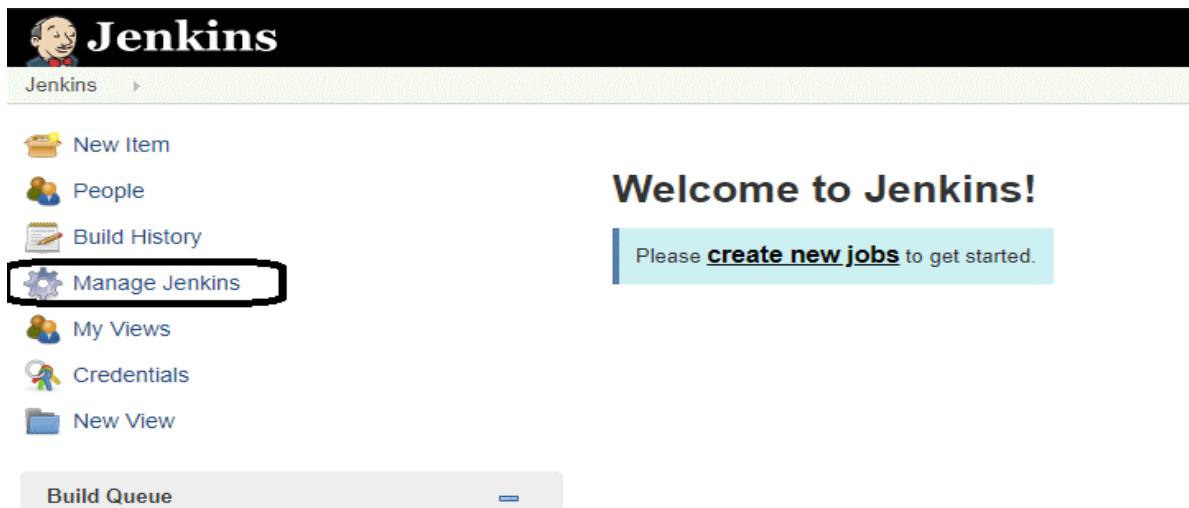
You can:

- Move a job from one installation of Jenkins to another by simply copying the corresponding job directory.
- Make a copy of an existing job by making a clone of a job directory by a different name.
- Rename an existing job by renaming a directory. Note that if you change a job name you will need to change any other job that tries to call the renamed job.

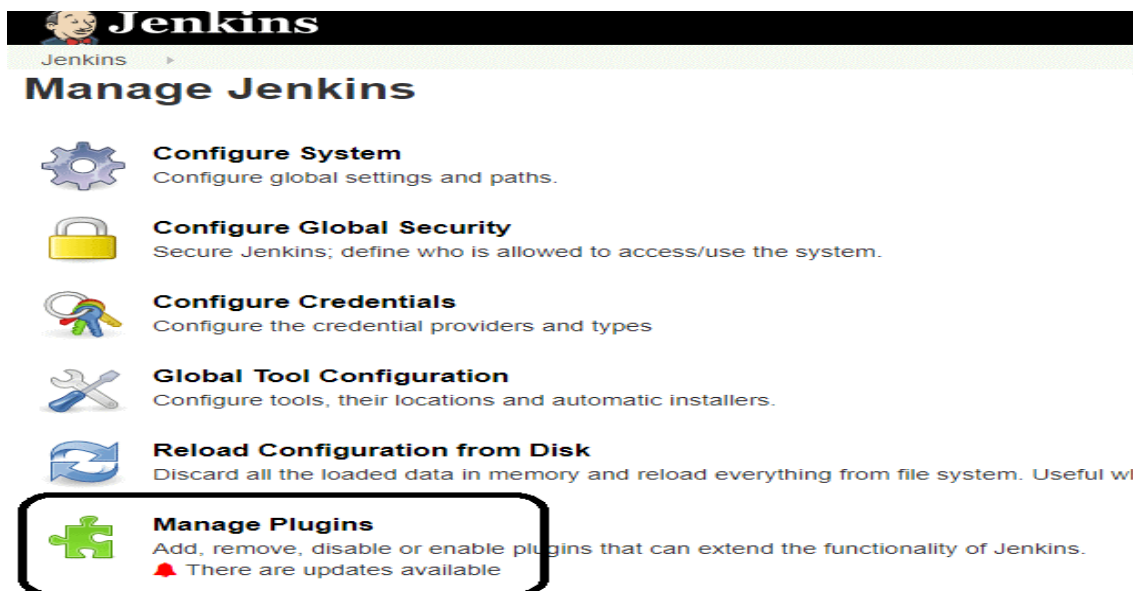
Q48. How do you integrate Git with Jenkins?

These are the steps to integrate [Git](#) with Jenkins –

1. Click on the **Manage Jenkins** button on your Jenkins dashboard:



2. Click on **Manage Plugins**.



3. In the Plugins Page

1. Select the GIT Plugin
2. Click on **Install without restart**. The plugin will take a few moments to finish downloading depending on your internet connection, and will be installed automatically.
3. You can also select the option **Download now and Install after restart**.

4. You will now see a “No updates available” message if you already have the Git plugin installed.

4. Once you install the plugins , go to **Manage Jenkins** on your Jenkins dashboard. You will see your plugins listed among the rest.

<input checked="" type="checkbox"/>	GitHub plugin This plugin integrates GitHub to Jenkins.	1.29.1	Uninstall
<input checked="" type="checkbox"/>	GitHub Branch Source Plugin Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.	2.3.6	Uninstall
<input checked="" type="checkbox"/>	GitHub API Plugin This plugin provides GitHub API for other plugins.	1.92	Uninstall
<input checked="" type="checkbox"/>	GIT server Plugin Allows Jenkins to act as a Git server.	1.7	Uninstall
<input checked="" type="checkbox"/>	Git plugin This plugin integrates Git with Jenkins.	3.9.1	Uninstall
<input checked="" type="checkbox"/>	Git client plugin Utility plugin for Git support in Jenkins	2.7.2	Uninstall

Q50. How can you temporarily turn off Jenkins security if the administrative users have locked themselves out of the admin console?

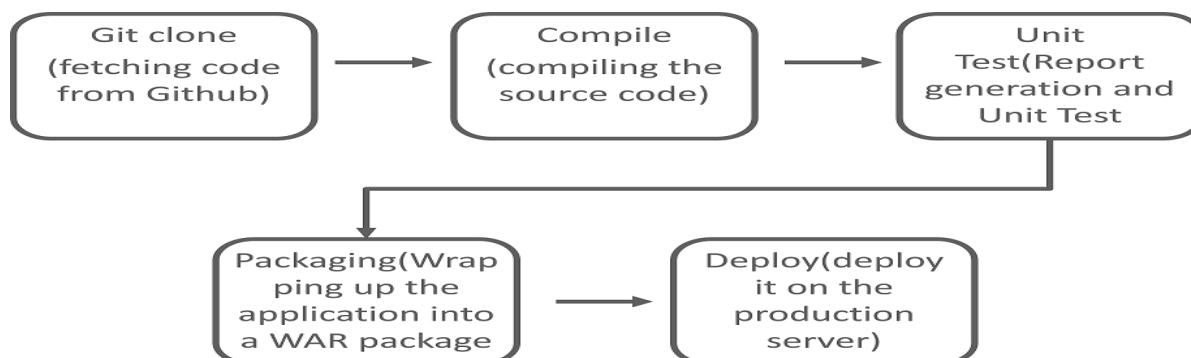
The JENKINS_HOME folder contains a file named config.xml. When you enable the security, this file contains an XML element named useSecurity that changes to true. If you change this setting to false, security will be disabled the next time Jenkins is restarted.

```
<useSecurity>false</useSecurity>
```

However, we must understand that disabling security should always be both a last resort and a temporary measure. Once you resolve the authentication issues, make sure that you re-enable Jenkins security and reboot the CI server.

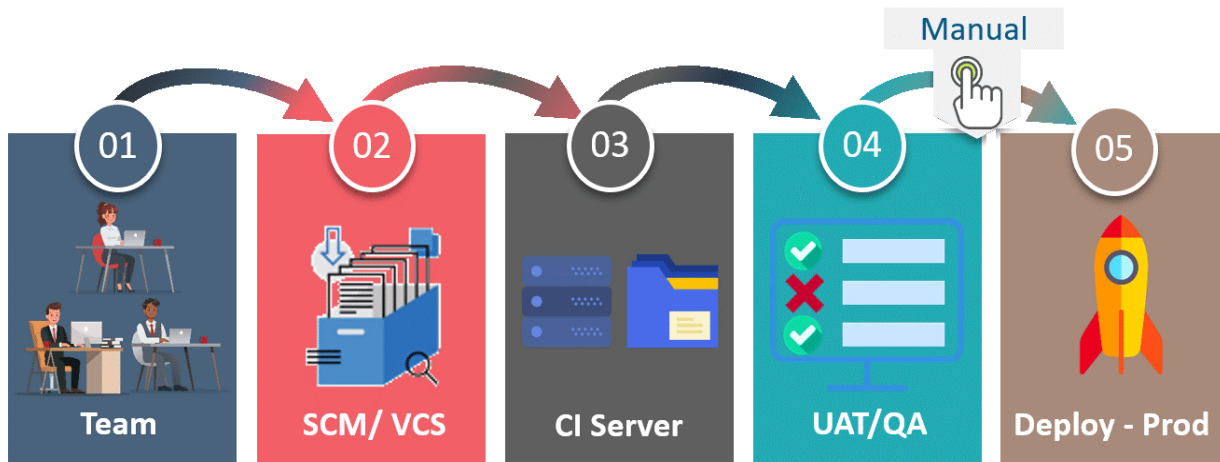
Q51. Can you define a Continuous Delivery Workflow?

The flowchart below shows the Continuous Delivery Workflow. Hope it will be much easier to understand with visuals.



Q52. What is the difference between Continuous Delivery and Continuous Deployment?

Continuous Delivery: (Manual Deployment to Production. Does not involve every change to be deployed.)



Continuous Delivery is a software development practice where you build software in such a way that the software can be released to the production at any time. You achieve Continuous Delivery by continuously integrating the products built by the development team, running automated tests on those built products to detect problems and then push those files into production-like environments to ensure that the software works in production.

Continuous Deployment: (Automated Deployment to Production. Involves deploying every change automatically)

Continuous deployment means that every change that you make, goes through the pipeline, and if it passes all the tests, it automatically gets deployed into production. So, with this approach, the quality of the software release completely depends on the quality of the test suite as you have automated everything.

Q53. What do you mean by Pipeline as a Code?

Pipeline as Code describes a set of features that allow Jenkins users to define pipelined job processes with code, stored and versioned in a source repository. These features allow Jenkins to discover, manage, and run jobs for multiple source repositories and branches — eliminating the need for manual job creation and management.

To use Pipeline as Code, projects must contain a file named **Jenkinsfile** in the repository root, which contains a “Pipeline script.”

Additionally, one of the enabling jobs needs to be configured in Jenkins:

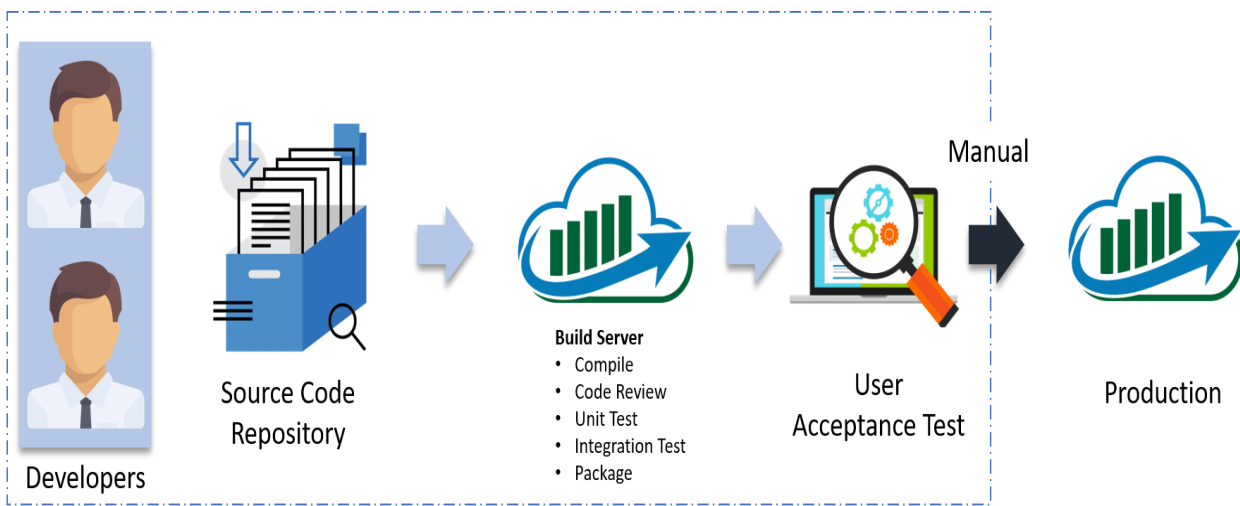
- **Multibranch Pipeline:** build multiple branches of a *single* repository automatically
- **Organization Folders:** scan a **GitHub Organization** or **Bitbucket Team** to discover an organization’s repositories, automatically creating managed Multibranch Pipeline jobs for them

ANSIBLE INTERVIEW QUESTIONS

Q1. What is CI/CD?

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually, each person integrates at least daily leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.

Continuous Delivery is a process where you build software in such a way that it can be released to production at any time. Consider the diagram below:



Let me explain the above diagram:

- Automated build scripts will detect changes in Source Code Management (SCM) like Git.
- Once the change is detected, source code would be deployed to a dedicated build server to make sure build is not failing and all test classes and integration tests are running fine.
- Then, the build application is deployed on the test servers (pre-production servers) for User Acceptance Test (UAT).
- Finally, the application is manually deployed on the production servers for release.

Q2. What is Configuration Management and how does it help an organization?

Configuration Management is the practice of handling updates and changes systematically so that a system maintains its integrity over time. Configuration Management (CM) keeps a track of all the updates that are needed in a system and it ensures that the current design and build state of the system is up to date and functioning correctly.

Configuration Management can help an 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.

Q3. 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:

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

Q4. How does Ansible work?

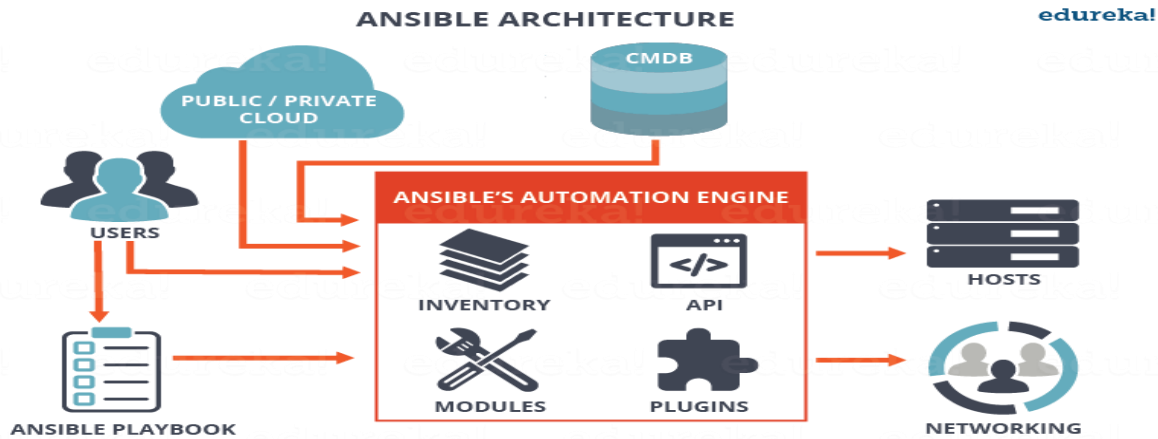
Ansible, unlike other configuration management tools, is categorized into two types of servers – Controlling machines and Nodes. Controlling machine is where Ansible is installed and nodes are the ones that are managed by the controlling machines through SSH. There is an inventory file in the controlling 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.

Q5. How is Ansible different from Puppet?

Metrics	Ansible	Puppet
1. Availability	• Highly available	• Highly available
2. Ease of set up	• Easy	• Comparatively hard to set up
3. Management	• Easy management	• Not very easy
4. Scalability	• Highly scalable	• Highly scalable
5. Configuration language	• YAML(Python)	• DSL(PuppetDSL)
6. Interoperability	• High	• High
7. Pricing nodes	• \$10,000	• \$11200-\$19900

Q6. What are the different components of ansible? Explain Ansible architecture.

The below diagram depicts the Ansible architecture:



Ansible Architecture – Ansible Interview Questions – Edureka

The main component of Ansible is the Ansible automation engine. This engine directly interacts with various cloud services, Configuration Management Database (CMDB) and different users who write various playbooks to execute the Ansible Automation engine.

The Ansible Automation engine consists of the following components:

Inventories: These are a list of nodes containing their respective IP addresses, servers, databases, etc. which needs to be managed.

APIs: Just like any other API, the Ansible APIs are used for commuting various Cloud services, public or private services.

Modules: The modules are used to manage system resources, packages, libraries, files, etc. Ansible modules can be used to automate a wide range of tasks. Ansible provides around 450 modules that automate nearly every part of your environment.

Plugins: If you want to execute Ansible tasks as a job, Ansible Plugins can be used. They simplify the execution of a task by building a job like an environment that basically contains pieces of code corresponding to some specific functionality. There are 100s of Plugins provided by Ansible. An example is the Action plugin, which acts as front ends to modules and can execute tasks on the controller before calling the modules themselves.

Networking: Ansible can also be used to automate different networks and services. It can do this by creating a playbook or an Ansible role that easily spans different network hardware.

Hosts: The Ansible Hosts/ Node systems are machines (Linux, Windows, etc) that are getting automated.

Playbooks: Playbooks are simple code files which describe the tasks that need to be executed. The Playbooks are written in YAML format. They can be used to automate tasks, declare configurations, etc.

CMDB: It is a database that acts as a storehouse for various IT installations. It holds data about various IT assets (also known as configuration items (CI)) and describes the relationships between such assets.

Cloud: It is a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server.

Q7. What are Ansible Server requirements?

If you are a windows user then you need to have a virtual machine in which Linux should be installed. It requires Python 2.6 version or higher.

Q8. How would you install Ansible on a CentOS system?

This can be done in two simple steps:

Step 1: Set up EPEL Repository

EPEL (Extra Packages for Enterprise Linux) is an open source and free community-based repository project from Fedora team which provides high-quality add-on software packages for Linux distribution including RHEL (Red Hat Enterprise Linux), CentOS, and Scientific Linux.

The Ansible package is not available in the default yum repositories, so we will enable EPEL repository by using the below command:

```
sudo rpm -ivh http://dl.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm
```

This will download all the necessary packages which will be required to install Ansible.

Step 2: Install Ansible

Now that your EPEL repository has been added, all you have to do now is install Ansible using the command below:

```
yum install ansible -y
```

To check the version of Ansible installed on your system, use the command below:

```
ansible --version
```

Q9. 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.

Q10. 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.

Q11. Compare Ansible with Chef.

Metrics	Ansible	Chef
1. Availability	• Highly available	• Highly available
2. Ease of set up	• Easy	• Not very easy
3. Management	• Easy management	• Not very easy
4. Scalability	• Highly scalable	• Highly scalable
5. Configuration language	• YAML(Python)	• DSL(Ruby)
6. Interoperability	• High	• High
7. Pricing nodes	• \$10,000	• \$13700

Q12. What is Ansible Galaxy?

Galaxy is a website that lets Ansible users share their roles and modules. The Ansible Galaxy command line tool comes packed with Ansible, and it can be used to install roles from Galaxy or directly from a Source Control Management system such as Git. It can also be used to build new roles, remove existing ones and perform tasks on the Galaxy website.

You can use the below command to download roles from the Galaxy website:

\$ansible-galaxy install username.role_name

Q13. What are Ad-hoc commands? Give an example.

Ad-hoc commands are simple one-line commands used to perform a certain task. You can think of Adhoc commands as an alternative to writing playbooks. An example of an Adhoc command is as follows:

```
ansible host -m netcaler -a "nsc_host=nsc.example.com user=apiuser password=apipass"
```

The above Adhoc command accesses the netcaler module to disable the server.

Q14. What are the variables in Ansible?

Variables in Ansible are very similar to variables in any programming language. Just like any other variable, an Ansible variable is assigned a value which is used in computing playbooks. You can also use conditions around the variables. Here's an example:

```
1 - hosts: your hosts
2 vars:
3   port_Tomcat : 8080
```

Here, we've defined a variable called port_Tomcat and assigned the port number 8080 to it. Such a variable can be used in the Ansible Playbook.

Q15. What is the difference between a variable name and an environment variable?

Variable name	Environment variable
<ul style="list-style-type: none">• You need to add strings to create variable names• You can easily create multiple variable names by adding strings• We use the ipv4 address for variable names	<ul style="list-style-type: none">• You need existing variables to access environment variables.• To create environment variables we must refer advanced Ansible playbook• We use {{ ansible_env.SOME_VARIABLE }} for remote environment variables.

Q16. What are the Ansible Modules? Explain the different types.

Ansible modules are a small set of programs that perform a specific task. Modules can be used to automate a wide range of tasks. Modules in Ansible are considered to be idempotent or in other words, making multiple identical requests has the same effect as making a single request.

There are 2 types of modules in Ansible:

1. Core modules
2. Extras modules

Core Modules

These are modules that the core Ansible team maintains and will always ship with Ansible itself. They will also receive a slightly higher priority for all requests than those in the "extras" repos. The source of these modules is hosted by Ansible on GitHub in the Ansible-modules-core.

Extras Modules

These modules are currently shipped with Ansible but might be shipped separately in the future. They are also mostly maintained by the Ansible Community. Non-core modules are still fully usable but may receive slightly lower response rates for issues and pull requests.

Popular “extras” modules may be promoted to core modules over time. The source for these modules is hosted by Ansible on GitHub in the Ansible-modules-extras.

Q17. What is Ansible Task?

Ansible Tasks allow you to break up bits of configuration policy into smaller files. These are blocks of code that can be used to automate any process. For example, if you wish to install a package or update a software, you can follow the below code snippet:

Install `<package_name>`, update `<software_name>`

Q18. Can you explain what are playbooks in Ansible? Explain with some examples.

Playbooks in Ansible are written in the YAML format. It is a human-readable data serialization language. It is commonly used for configuration files. It can also be used in many applications where data is being stored.

For Ansible, nearly every YAML file starts with a list. Each item in the list is a list of key/value pairs, commonly called a “hash” or a “dictionary”. So, we need to know how to write lists and dictionaries in YAML.

All members of a list are lines beginning at the same indentation level starting with a “- ” (dash and space). More complicated data structures are possible, such as lists of dictionaries or mixed dictionaries whose values are lists or a mix of both.

For example, if you want a playbook containing details about the USA:

```
1  -USA
2  -continent: North America
3  -capital: Washington DC
4  -population: 319 million
```

Now that you know the basic level questions, let’s take a look at a little more advanced Ansible Interview Questions.

Intermediate Level Ansible Interview Questions

Once you’ve aced the basic conceptual questions, the interviewer will increase the difficulty level. So let’s move on to the next section of this Ansible Interview Questions article. This section talks about the commands that are very common amongst docker users.

Q19. Can you write a simple playbook to install Nginx on a host machine?

Step 1: Generate a public SSH key and by using SSH connect to your host.

Follow the command below:

\$ ssh-keygen

```
File Edit View Search Terminal Help
[edureka@edureka ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/edureka/.ssh/id_rsa):
/home/edureka/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/edureka/.ssh/id_rsa.
Your public key has been saved in /home/edureka/.ssh/id_rsa.pub.
The key fingerprint is:
7e:cf:4e:6e:5f:c2:43:1c:f4:8f:6e:44:35:ab:d5:de edureka@edureka
The key's randomart image is:
+--[ RSA 2048 ]-----+
|           ...      |
|          . 0+      |
|         ooo        |
|        oo+o        |
|       S  . = E      |
|      .  =           |
|     . . . * .       |
|    . = . . +        |
|   o = . .           |
+-----+
[edureka@edureka ~]$
```

Ansible Playbook – Ansible Interview Questions – Edureka

As shown above, a public SSH key is generated.

Step 2: Next, copy the public SSH key on your hosts. Follow the below command to do it:

ssh-copy-id -i root@IP address of your host

```
[root@edureka edureka]# ssh-copy-id -i root@10.0.2.15
The authenticity of host '10.0.2.15 (10.0.2.15)' can't be established.
RSA key fingerprint is 2d:af:5f:52:24:59:05:f0:1e:12:60:71:cb:5d:8f:80.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.15' (RSA) to the list of known hosts.
root@10.0.2.15's password:
Now try logging into the machine, with "ssh 'root@10.0.2.15'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

[root@edureka edureka]#
```

Ansible Playbook – Ansible Interview Questions – Edureka

Step 3: List the IP addresses of your hosts/nodes in your inventory.

Follow the below command:

vi /etc/ansible/hosts

```
## db-[99:101]-node.example.com
[test-server]
172.17.0.2
INCBT
```

Ansible Playbook – Ansible Interview Questions – Edureka

Once you run the command, the vi editor will open where you can list down the IP addresses of your hosts. This is now your inventory.

Step 4: To check if the connection has been established, let's ping:

```
File Edit View Search Terminal Help
[root@edureka edureka]# ansible -m ping 'test-server'
10.0.2.15 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
[root@edureka edureka]#
```

Ansible Playbook – Ansible Interview Questions – Edureka

The above image shows that a connection has been made between the control machine and the host.

Step 5: Create a playbook to install Nginx on the host machine. To create a playbook you just need to open a file with a yaml extension, like shown below:

vi <Name of your file>.yaml

```
File Edit View Search Terminal Help
--
hosts: test-server
  sudo: yes
  vars:
    - server_port: 8080

tasks:
  - name: install nginx
    yum: pkg=nginx state=installed

  - name: serve nginx config
    template: src=../files/flask.conf dest=/etc/nginx/conf.d/
    notify:
      - restart nginx

handlers:
  - name: restart nginx
    service: name=nginx state=restarted
```

Ansible Playbook – Ansible Interview Questions – Edureka

In an Ansible Playbook, the tasks are defined as a list of dictionaries and are executed from top to bottom.

Each task is defined as a dictionary that can have several keys, such as “name” or “sudo” which signify the name of the task and whether it requires sudo privileges.

A variable `server_port` is set that listens on TCP port 8080 for incoming requests.

Here, the first task is to get the necessary package for installation of Nginx and then install it. Internally, Ansible will check if the directory exists and create it if it’s not, otherwise, it will do nothing.

The next task is to configure Nginx. In Nginx, contexts contain configuration details.

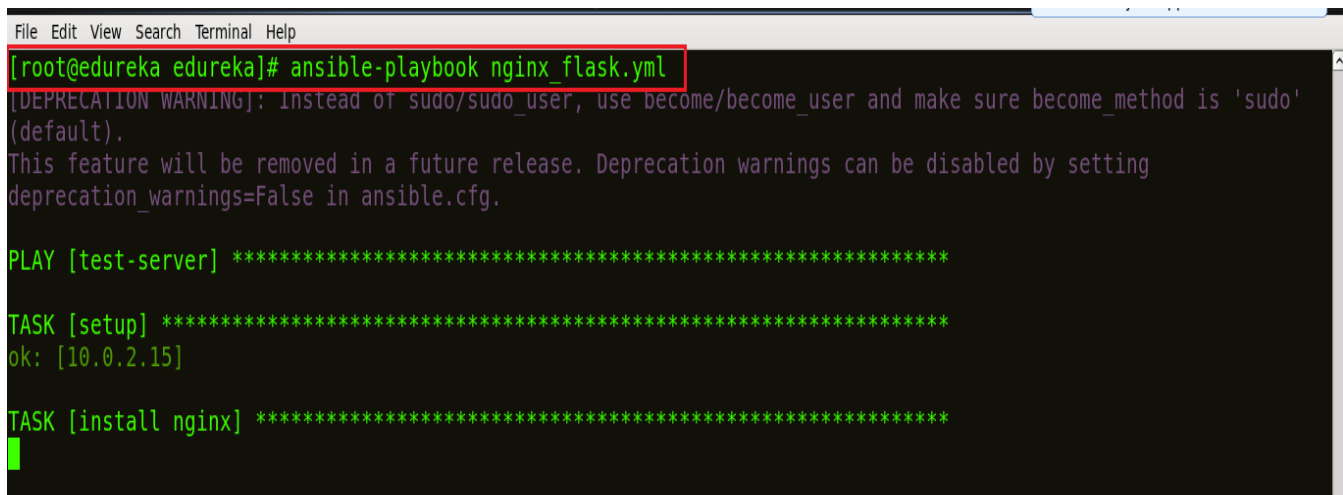
Here, the template is a file you can deploy on hosts. However, template files also include some reference variables which are pulled from variables defined as part of an Ansible playbook or facts gathered from the hosts. Facts containing the configuration details are being pulled from a source directory and being copied to a destination directory.

Handlers here define the action to be performed only upon notification of tasks or state changes. In this playbook, we defined, notify: restart Nginx handler which will restart Nginx once the files and templates are copied to hosts.

Now, save the file and exit.

Step 6: Run the playbook, using the command below:

`ansible-playbook <name of your file>.yaml`



```
File Edit View Search Terminal Help
[root@edureka edureka]# ansible-playbook nginx_flask.yml
[DEPRECATION WARNING]: Instead of sudo/sudo_user, use become/become_user and make sure become_method is 'sudo'
(default).
This feature will be removed in a future release. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.

PLAY [test-server] *****

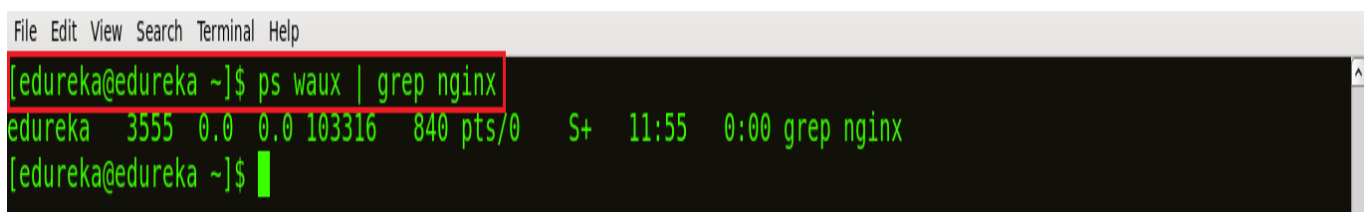
TASK [setup] *****
ok: [10.0.2.15]

TASK [install nginx] *****
```

Ansible Playbook – Ansible Interview Questions – Edureka

Step 7: Check if Nginx is installed on the machine. Use the following command:

`ps waux | grep nginx`



```
File Edit View Search Terminal Help
[edureka@edureka ~]$ ps waux | grep nginx
edureka 3555 0.0 0.0 103316 840 pts/0 S+ 11:55 0:00 grep nginx
[edureka@edureka ~]$
```


In the above image, the different process IDs 3555 and 103316 are running which shows that Nginx is running on your host machines.

Q20. How would you access a variable of the first host in a group?

This can be done by executing the below command:

```
{{ hostvars[groups['webservers'][0]]['ansible_eth0']['ipv4']['address'] }}
```

In the above command, we're basically accessing the hostname of the first machine in the webservers group. If you're using a template to do this, use the Jinja2 '#set' or you can also use set_fact, like shown below:

```
1 - set_fact: headnode={{ groups['webservers'][0] }}
2 - debug: msg={{ hostvars[headnode].ansible_eth0.ipv4.address }}
```

Q21. Why is '{{ {{ }} }}' notation used? And how can one interpolate variables or dynamic variable names?

One basic rule is to 'always use {{ {{ }} }} except when:'. Conditionals are always run through Jinja2 as to resolve the expression. Therefore, 'when:failed_when:' and 'changed_when:' are always templated and we should avoid adding {{ {{ }} }}. In other cases, except when clause, we have to use brackets, otherwise, differentiating between an undefined variable and a string will be difficult to do.

Q22. What is Ansible role and how are they different from the playbook?

[Ansible Roles](#) is basically another level of abstraction used to organize playbooks. They provide a skeleton for an independent and reusable collection of variables, tasks, templates, files, and modules which can be automatically loaded into the playbook. Playbooks are a collection of roles. Every role has specific functionality.

Let's understand the difference between Ansible roles and playbook with an example.

Suppose you want your playbook to perform 10 different tasks on 5 different systems, would you use a single playbook for this? No, using a single playbook can make it confusing and prone to blunders. Instead, you can create 10 different roles, where each role will perform one task. Then, all you need to do is, mention the name of the role inside the playbook to call them.

Q23. How do I write an Ansible handler with multiple tasks?

Suppose you want to create a handler that restarts a service only if it is already running.

Handlers can "listen" to generic topics, and tasks can notify those topics as shown below. This functionality makes it much easier to trigger multiple handlers. It also decouples handlers from their names, making it easier to share handlers among playbooks and roles:


```
1  - name: Check if restarted
2  shell: check_is_started.sh
3  register: result
4  listen: Restart processes
5
6
7
8  - name: Restart conditionally step 2
9  service: name=service state=restarted
10 when: result
11 listen: Restart processes
```

Q24. How to keep secret data in a playbook?

Suppose you have a task that you don't want to show the output or command given to it when using -v (verbose) mode, the following task can be used to do it:

```
1  - name: secret task
2  shell: /usr/bin/do_something --value={{ secret_value }}
3  no_log: True
```

This can be used to keep verbose output but hide sensitive information from others who would otherwise like to be able to see the output.

The no_log attribute can also apply to an entire play:

```
1  - hosts: all
2  no_log: True
```

Q25. What are Ansible Vaults and why are they used?

[Ansible Vault](#) is a feature that allows you to keep all your secrets safe. It can encrypt entire files, entire YAML playbooks or even a few variables. It provides a facility where you can not only encrypt sensitive data but also integrate them into your playbooks.

Vault is implemented with file-level granularity where the files are either entirely encrypted or entirely unencrypted. It uses the same password for encrypting as well as for decrypting files which makes using Ansible Vault very user-friendly.

Q26. How to create encrypted files using Ansible?

To create an encrypted file, use the 'ansible-vault create' command and pass the filename.

```
$ ansible-vault create filename.yaml
```

You'll be prompted to create a password and then confirm it by re-typing it.

```
ubuntu@ip-10-0-2-54:~$ ansible-vault create secrets.txt
New Vault password:
Confirm New Vault password:
```

Ansible Playbook – Ansible Interview Questions – Edureka

Once your password is confirmed, a new file will be created and will open an editing window. By default, the editor for Ansible Vault is vi. You can add data, save and exit.

Ansible Playbook – Ansible Interview Questions – Edureka

This is your encrypted file:

```
ubuntu@ip-10-0-2-54:~$ cat secrets.txt
$ANSIBLE_VAULT;1.1;AES256
64633735613636656665343436316337626635316161323130323236343039303935656132613937
3031353664346635613431616631663731313339346231390a343233666163633433613232613631
323531633313033323739656664376536333038633038326639653738333435383961666233333661
3633363037366533610a663565646230353239353462333338623164393361386431316330343962
65643839663737623134306366653239626636303866323030323634656365306165373730353935
31333465323839656564633464663962386666663130373032396363323863633936316264663439
653764323731653964653332366564633739
```

Ansible Playbook – Ansible Interview Questions – Edureka

Q27. What is Ansible Tower?

[Ansible Tower](#) is Ansible at a more enterprise level. It is a web-based solution for managing your organization with a very easy user interface that provides a dashboard with all of the state summaries of all the hosts, allows quick deployments, and monitors all configurations.

The tower allows you to share the SSH credentials without exposing them, logs all the jobs, manage inventories graphically and syncs them with a wide variety of cloud providers.

Q28. What features does the Ansible Tower provide?

- **Ansible Tower Dashboard** – The Ansible Tower dashboard displays everything going on in your Ansible environment like the hosts, inventory status, the recent job activity and so on.
- **Real-Time Job Updates** – As Ansible can automate the complete infrastructure, you can see real-time job updates, like plays and tasks broken down by each machine either been successful or a failure. So, with this, you can see the status of your automation, and know what's next in the queue.
- **Multi-Playbook Workflows** – This feature allows you to chain any number of playbooks, regardless of the usage of different inventories, utilizes various credentials, or runs different users.

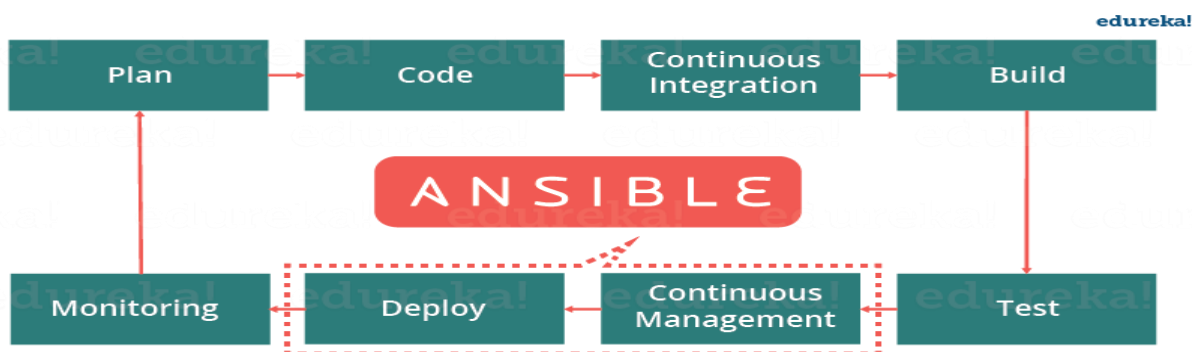
- **Who Ran What Job When** – As the name suggests, you can easily know who ran what job where and when as, all the automation activity is securely logged in Ansible Tower.
- **Scale Capacity With Clusters** – We can connect multiple Ansible Tower nodes into an Ansible Tower cluster as the clusters add redundancy and capacity, which allow you to scale Ansible automation across the enterprise.
- **Integrated Notifications** – This feature lets you notify a person or team when a job succeeds or fails across the entire organization at once, or customize on a per-job basis.
- **Schedule Ansible Jobs** – Different kinds of jobs such as Playbook runs, cloud inventory updates, and source control updates can be scheduled inside Ansible Tower to run according to the need.
- **Manage & Track Inventory** – Ansible Tower helps you manage your entire infrastructure by letting you easily pull inventory from public cloud providers such as Amazon Web Services, Microsoft Azure, and more.
- **Self-Service** – This feature of Ansible Tower lets you launch Playbooks with just a single click. It can also, let you choose from available secure credentials or prompt you for variables and monitor the resulting deployments.
- **REST API & Tower CLI Tool** – Every feature present in Ansible Tower is available via Ansible Tower's REST API, which provides the ideal API for a systems management infrastructure. The Ansible Tower's CLI tool is available for launching jobs from CI systems such as Jenkins, or when you need to integrate with other command-line tools.
- **Remote Command Execution** – You can run simple tasks such as add users, restart any malfunctioning service, reset passwords on any host or group of hosts in the inventory with Ansible Tower's remote command execution.

The following section will cover only the technical based Ansible Interview Questions. These questions are mostly based on the practical implementation of Ansible.

Ansible Technical Interview Questions

Q29. How is Ansible used in a Continuous Delivery pipeline? Explain.

It is well known that in DevOps development and operations work is integrated. This integration is very important for modern test-driven applications. Hence, Ansible integrates this by providing a stable environment to both development and operations resulting in a smooth delivery pipeline.



When developers begin to think of infrastructure as part of their application i.e as Infrastructure as code (IaC), stability and performance become normative. Infrastructure as Code is the process of managing and provisioning computing infrastructure and their configuration through machine-processable definition files, rather than physical hardware configuration or the use of interactive configuration tools. This is where Ansible automation plays a major role and stands out among its peers.

In a Continuous Delivery pipeline, Sysadmins work tightly with developers, development velocity is improved, and more time is spent doing activities like performance tuning, experimenting, and getting things done, and less time is spent fixing problems.

Q30. How can you create a LAMP stack and deploy a webpage by using Ansible?

Suppose you're trying to deploy a website on 30 systems, every website deployment will require a base OS, web-server, Database, and PHP. We use ansible playbook to install these prerequisites on all 30 systems at once.

For this particular problem statement, you can use two virtual machines, one as a server where Ansible is installed and the other machine acts as the remote host. Also, I've created a simple static webpage saved in a folder index which has two files, index.html, and style.css.

In the below code I've created a single Ansible playbook to install Apache, MySql, and PHP:

```
1  ---
2  # Setup LAMP Stack
3  - hosts: host1
4  tasks:
5
6  - name: Add ppa repository
7    become: yes
8    apt_repository: repo=ppa:ondrej/php
9
10 - name: Install lamp stack
11   become: yes
12   apt:
13     pkg:
14     - apache2
15     - mysql-server
16     - php7.0
```

```
17 - php7.0-mysql
18 state: present
19 update cache: yes
20
21 - name: start apache server
22 become: yes
23 service:
24 name: apache2
25 state: started
26 enabled: yes
27
28 - name: start mysql service
29 become: yes
30 services:
31 name: mysql
32 state: started
33 enabled: yes
34
35 - name: create target directory
36 file: path=/var/www/html state=directory mode=0755
37
38 - name: deploy index.html
39 become: yes
40 copy:
41 src: /etc/ansible/index/index.html
42 dest: var/www/html/index/index.html
```

Now, there are 6 main tasks, each task performs a specific function:

- The first task adds the repository required to install MySQL and PHP.
- The second task installs apache2, MySQL-server, PHP, and PHP-MySQL.
- The third and fourth task starts the Apache and MySQL service.
- The fifth task creates a target directory in the host machine and

- Finally, the sixth task executes the index.html file, it picks up the file from the server machine and copies it into the host machine.

To finally run this playbook you can use the following command:

```
$ ansible-playbook lamp.yml -K
```

Q31. How do I set the PATH or any other environment variable for a task?

The environment variables can be set by using the 'environment' keyword. It can be set for either a task or an entire playbook as well. Follow the below code snippet to see how:

```
1  environment:
2  PATH: "{{ ansible_env.PATH }}:/thingy/bin"
3  SOME: value
```

Q32. How can one generate encrypted passwords for the user module?

There are a couple of ways to do this. The simplest way is to use the ad-hoc command:

```
ansible all -i localhost, -m debug -a "msg={{ 'mypassword' | password_hash('sha512', 'mysecretsalt') }}"
```

Another way is to use the mkpasswd functionality available on Linux systems:

```
mkpasswd --method=sha-512
```

However, if you're using a macOS then you can generate these passwords using Python. To do this you must first install the Passlib password hashing library:

```
pip install passlib
```

After installing it, the SHA512 password values can be generated in the following manner:

```
python -c "from passlib.hash import sha512_crypt; import getpass; print(sha512_crypt.using(rounds=5000).hash(getpass.getpass()))"
```

Q33. How can looping be done over a list of hosts in a group, inside of a template?

An easy way to do this is to iterate over a list of hosts inside of a host group, in order to fill a template configuration file with a list of servers. This can be done by accessing the "\$groups" dictionary in your template, like so:

```
In 1  {% for host in groups['db_servers'] %}
   2  {{ host }}
   3  {% endfor %}
```

order to access facts about these hosts, like, the IP address of each hostname, you need to make sure that the facts have been populated. For instance, make sure you have a play that talks to db_servers:

```

1 - hosts: db_servers
2 tasks:
3 - debug: msg="doesn't matter what you do, just that they were talked to
  previously."

```

Now you can use the facts within your template, like so:

```

1 { % for host in groups['db_servers'] % }
2 { { hostvars[host]['ansible_eth0']['ipv4']['address'] } }
3 { % endfor % }

```

Q34. How can I display all the inventory vars defined for my host?

In order to check the inventory vars resulting from what you've defined in the inventory, you can execute the below command:

```
ansible -m debug -a "var=hostvars['hostname']" localhost
```

This will list down all the inventory vars.

Q35. How should one configure a jump host to access servers that I have no direct access to?

The first step would be to set a ProxyCommand in the ansible_ssh_common_args inventory variable. All arguments that are defined in this variable are added to the sftp/scp/ssh command line when connecting to the relevant host. Let's look at

an example, consider the below inventory group:

```

1 [gatewayed]
2 foo ansible_host=192.0.2.1
3 bar ansible_host=192.0.2.2

```

Next, you can create group_vars/gatewayed.yml containing the following:

```
ansible_ssh_common_args: '-o ProxyCommand="ssh -W %h:%p -q user@gateway.example.com"'
```

Ansible will then append these arguments to the command line while trying to connect to any hosts in the group gatewayed.

Q36. How can you handle different machines needing different user accounts or ports to log in with?

The simplest way to do this is by setting inventory variables in the inventory file.

Let's consider that these hosts have different usernames and ports:


```
1 [webservers]
2 asdf.example.com ansible_port=5000 ansible_user=alice
3 jkl.example.com ansible_port=5001 ansible_user=bob
```

Also, if you wish to, you can specify the connection type to be used:

```
1 [testcluster]
2 localhost ansible_connection=local
3 /path/to/chroot1 ansible_connection=chroot
4 foo.example.com ansible_connection=paramiko
```

To make this more clear it is best to keep these in group variables or file them in a `group_vars/<group-name>` file.

Q37. Is it unsafe to bulk-set task arguments from a variable?

To set all the arguments in a task you can use the dictionary-typed variable. Even though this is usually good for dynamic executions, it induces a security risk. Therefore, when this happens, Ansible issues a warning. For example, consider the below code:

```
1 vars:
2 usermod_args:
3   name: testuser
4   state: present
5 tasks:
6   - user: '{{ usermod_args }}
```

This example is safe but creating similar tasks is risky because the parameters and values passed to `usermod_args` could be overwritten by malicious values in the host facts on a compromised target machine.

Q38. Suppose you're using Ansible to configure the production environment and your playbook uses an encrypted file. Encrypted files prompt the user to enter passwords. But since Ansible is used for automation, can this process be automated?

Yes, Ansible uses a feature called password file, where all the passwords to your encrypted files can be saved. So each time the user is asked for the password, he can simply make a call to the password file. The password is automatically read and entered by Ansible.

\$ ansible-playbook launch.yml --vault-password-file ~/.vault_pass.txt

Having a separate script that specifies the passwords is also possible. You need to make sure the script file is executable and the password is printed to standard output for it to work without annoying errors.


```
$ ansible-playbook launch.yml --vault-password-file ~/.vault_pass.py
```

Q39. Have you worked with Ansible before? Please share your experience.

Be very honest here. If you have used ansible before then talk about your experience. Talk about the projects that required ansible. You can tell the interviewer about how Ansible has helped you in provisioning and configuration management. If you haven't used Ansible before then just talk about any related tools that you've used. These related tools could be anything like Git, Jenkins, Puppet, Chef, Saltstack, etc.

Be very honest because they know if you're lying.

Q40. Is Ansible an Open Source tool?

Yes, Ansible is open source. That means you take the modules and rewrite them. Ansible is an open-source automated engine that lets you automate apps.

Q41. How can you connect other devices within Ansible?

Once Ansible is installed on the controlling machines, an inventory file is created. This inventory file specifies the connection between other nodes. A connection can be made using a simple SSH. To check the connection to a different device, you can use the ping module.

```
ansible -m ping all
```

The above command checks the connection to all the nodes specified in the inventory file.

Q42. Is it possible to build our modules in Ansible?

Yes, we can create our own modules within Ansible. It's an open-source tool which basically works on python. You can start creating your own modules. The only requirements would be to be amazingly good at programming.

Q43. What does Fact mean in Ansible?

When any new variable about the system has been discovered it's considered to be a "fact" in the playbook. Facts are mainly used to implement conditional executions. It can also be used to get the ad-hoc information about the system.

You can get facts with the following command:

```
$ ansible all- m setup
```

So when you want to extract only a part of the information, you use the setup module to filter out only the needed information.

Q44. What is the ask_pass module in Ansible?

Ask_pass is the control module in an Ansible playbook. This controls the prompting of the password when the playbook is getting executed. By default, it's always set to True. If you are using SSH keys for authentication purposes then you really don't have to change this setting at all.

Q45. Explain the callback plugin in Ansible?

Callback plugins are enable adding new behaviors to Ansible when responding to events. By default, callback plugins control most of the output you see when running the command line program. It can also be used to add additional output, integrate with other tools, etc.

Q46. Does Ansible support AWS?

Ansible has hundreds of modules supporting AWS and some of them include:

- Autoscaling groups
- CloudFormation
- CloudTrail
- CloudWatch
- DynamoDB
- ElastiCache
- Elastic Cloud Compute (EC2)
- Identity Access Manager (IAM)
- Lambda
- Relational Database Service (RDS)
- Route53
- Security Groups
- Simple Storage Service (S3)
- Virtual Private Cloud (VPC)

Q47. Does Ansible support hardware provisioning?

Yes, Ansible can provision hardware. A lot of companies are still stuck on to massive data centers of hardware. There are a few requirements. You must set up some services before you go ahead. Some of them are – DHCP, PXE, TFTP, Operating System Media, Web Server, etc.

Q48. Write an Ansible playbook to automate the starting of EC2 instance.

```
1 ---
2 - name: Create an ec2 instance
3   hosts: web
4   gather_facts: false
5
6   vars:
7     region: us-east-1
8     instance_type: t2.micro
9     ami: ami-05ea7729e394412c8
10    keypair: priyajdm
11
12   tasks:
13     - name: Create an ec2 instance
14       ec2:
15         aws_access_key: '*****'
16         aws_secret_key: '*****'
17         key_name: "{{ keypair }}"
18         group: launch-wizard-26
19         instance_type: "{{ instance_type }}"
20         image: "{{ ami }}"
21         wait: true
22         region: "{{ region }}"
23         count: 1
24         vpc_subnet_id: subnet-02f498e16fd56c277
25         assign_public_ip: yes
26         register: ec2
27
```

- We start by mentioning AWS access key id and secret key using the parameters **aws_access_key** and **aws-secret_key**.
- **key_name**: pass the variable that defines the keypair being used here

- **group:** mention the name of the security group. This defines the security rules of the EC2 instance we're trying to bring up
- **instance_type:** pass the variable that defines the type of instance we're using here
- **image:** pass the variable that defines the AMI of the image we're trying to start.
- **wait:** This has a boolean value of either true or false. If true, it waits for the instance to reach the desired state before returning
- **region:** pass the variable that defines the region in which an EC2 instance needs to be created.
- **count:** This parameter specifies the number of instances that need to be created. In this case, I've only mentioned only one but this depends on your requirements.
- **vpc_subnet_id:** pass the subnet id in which you wish to create the instance
- **assign_public_ip:** This parameter has a boolean value. If true like in our case, a public IP will be assigned to the instance when provisioned within VPC.

Q49. Can you copy files recursively onto a target host? If yes, how?

Yes, you can copy files recursively onto a target host using the copy module. It has a recursive parameter which copies files from a directory. There is another module called synchronize which is specifically made for this.

```
1  - synchronize:
2  src: /first/absolute/path
3  dest: /second/absolute/path
4  delegate_to: "{{ inventory_hostname }}"
```

Q50. Write a playbook to create a backup of a file in the remote servers before copy.

This is pretty simple. You can use the below playbook:

```
1  - hosts: blocks
2  tasks:
3  - name: ansible copy file backup example
4  copy:
5  src: ~/helloworld.txt
6  dest: /tmp
7  backup: yes
```

DOCKER INTERVIEW QUESTIONS

1. What is Hypervisor?

A hypervisor is a software that makes virtualization possible. It is also called Virtual Machine Monitor. It divides the host system and allocates the resources to each divided virtual environment. You can basically have multiple OS on a single host system. There are two types of Hypervisors:

- Type 1: It's also called Native Hypervisor or Bare metal Hypervisor. It runs directly on the underlying host system. It has direct access to your host's system hardware and hence does not require a base server operating system.
- Type 2: This kind of hypervisor makes use of the underlying host operating system. It's also called Hosted Hypervisor.

2. What is virtualization?

Virtualization is the process of creating a software-based, virtual version of something (compute storage, servers, application, etc.). These virtual versions or environments are created from a single physical hardware system. Virtualization lets you split one system into many different sections which act like separate, distinct individual systems. A software called Hypervisor makes this kind of splitting possible. The virtual environment created by the hypervisor is called Virtual Machine.

3. What is containerization?

Let me explain this with an example. Usually, in the software development process, code developed on one machine might not work perfectly fine on any other machine because of the dependencies. This problem was solved by the containerization concept. So basically, an application that is being developed and deployed is bundled and wrapped together with all its configuration files and dependencies. This bundle is called a container. Now when you wish to run the application on another system, the container is deployed which will give a bug-free environment as all the dependencies and libraries are wrapped together. Most famous containerization environments are Docker and Kubernetes.

4. Difference between virtualization and containerization?

Once you've explained containerization and virtualization, the next expected question would be differences. The question could either be differences between virtualization and containerization or differences between virtual machines and containers. Either way, this is how you respond.

Containers provide an isolated environment for running the application. The entire user space is explicitly dedicated to the application. Any changes made inside the container is never reflected on the host or even other containers running on the same host. Containers are an abstraction of the application layer. Each container is a different application.

Whereas in Virtualization, hypervisors provide an entire virtual machine to the guest(including Kernal). Virtual machines are an abstraction of the hardware layer. Each VM is a physical machine.

5. What is Docker?

Since its a Docker interview, there will be an obvious question about what is Docker. Start with a small definition.

Docker is a containerization platform which packages your application and all its dependencies together in the form of containers so as to ensure that your application works seamlessly in any environment, be it development, test or production. Docker containers, wrap a piece of software in a complete filesystem that contains everything needed to run: code, runtime, system tools, system libraries, etc. It wraps basically anything that can be installed on a server. This guarantees that the software will always run the same, regardless of its environment.

6. What is a Docker Container?

Docker containers include the application and all of its dependencies. It shares the kernel with other containers, running as isolated processes in user space on the host operating system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud. Docker containers are basically runtime instances of Docker images.

7. What are Docker Images?

When you mention Docker images, your very next question will be “what are Docker images”.

Docker image is the source of Docker container. In other words, Docker images are used to create containers. When a user runs a Docker image, an instance of a container is created. These docker images can be deployed to any Docker environment.

8. What is Docker Hub?

Docker images create docker containers. There has to be a registry where these docker images live. This registry is Docker Hub. Users can pick up images from Docker Hub and use them to create customized images and containers. Currently, the [Docker Hub](https://hub.docker.com/) is the world’s largest public repository of image containers.

9. Explain Docker Architecture?

Docker Architecture consists of a Docker Engine which is a client-server application with three major components:

1. A server which is a type of long-running program called a daemon process (the docker command).
2. A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
3. A command line interface (CLI) client (the docker command).
4. The CLI uses the Docker REST API to control or interact with the Docker daemon through scripting or direct CLI commands. Many other Docker applications use the underlying API and CLI.

10. What is a Dockerfile?

Docker can build images automatically by reading the instructions from a file called Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build, users can create an automated build that executes several command-line instructions in succession.

The interviewer does not just expect definitions, hence explain how to use a Dockerfile which comes with experience. Have a look at [this](#) tutorial to understand how Dockerfile works.

11. Tell us something about Docker Compose.

Docker Compose is a YAML file which contains details about the services, networks, and volumes for setting up the Docker application. So, you can use Docker Compose to create separate containers, host them and get them to communicate with each other. Each container will expose a port for communicating with other containers.

12. What is Docker Swarm?

You are expected to have worked with Docker Swarm as it's an important concept of Docker.

Docker Swarm is native clustering for Docker. It turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts.

13. What is a Docker Namespace?

A namespace is one of the Linux features and an important concept of containers. Namespace adds a layer of isolation in containers. Docker provides various namespaces in order to stay portable and not affect the underlying host system. Few namespace types supported by Docker – PID, Mount, IPC, User, Network

14. What is the lifecycle of a Docker Container?

This is one of the most popular questions asked in Docker interviews. Docker containers have the following lifecycle:

- Create a container
- Run the container
- Pause the container(optional)
- Un-pause the container(optional)
- Start the container
- Stop the container
- Restart the container
- Kill the container
- Destroy the container

15. What is Docker Machine?

Docker machine is a tool that lets you install Docker Engine on virtual hosts. These hosts can now be managed using the docker-machine commands. Docker machine also lets you provision Docker Swarm Clusters.

Docker Basic Commands

Once you've aced the basic conceptual questions, the interviewer will increase the difficulty level. So let's move on to the next section of this Docker Interview Questions article. This section talks about the commands that are very common amongst docker users.

16. How to check for Docker Client and Docker Server version?

```
$ docker version
```

17. How do you get the number of containers running, paused and stopped?

```
$ docker info
```

You can get the number of containers running, paused, stopped, the number of images and a lot more.

18. If you vaguely remember the command and you'd like to confirm it, how will you get help on that particular command?

```
$ docker --help
```

The above command lists all Docker commands. If you need help with one specific command, you can use the following syntax:

```
$ docker <command> --help
```

19. How to login into docker repository?

You can use the following command to login into hub.docker.com:

```
$ docker login
```

You'll be prompted for your username and password, insert those and congratulations, you're logged in.

20. If you wish to use a base image and make modifications or personalize it, how do you do that?

You pull an image from docker hub onto your local system

```
$ docker pull <image_name>
```

21. How do you create a docker container from an image?

Pull an image from docker repository with the above command and run it to create a container. Use the following command:

```
$ docker run -it -d <image_name>
```

Most probably the next question would be, what does the '-d' flag mean in the command?

-d means the container needs to start in the detached mode. Explain a little about the detach mode.

22. How do you list all the running containers?

```
$ docker ps
```

23. Suppose you have 3 containers running and out of these, you wish to access one of them. How do you access a running container?

```
$ docker exec -it <container id> bash
```

The exec command lets you get inside a container and work with it.

24. How to start, stop and kill a container?

The following command is used to start a docker container:

```
$ docker start <container_id>
```

and the following for stopping a running container:

```
$ docker stop <container_id>
```

kill a container with the following command:

```
$ docker kill <container_id>
```

25. Can you use a container, edit it, and update it? Also, how do you make it a new and store it on the local system?

Of course, you can use a container, edit it and update it. This sounds complicated but its actually just one command.

```
$ docker commit <container id> <username/imagename>
```

26. Once you've worked with an image, how do you push it to docker hub?

```
$ docker push <username/image name>
```

27. How to delete a stopped container?

```
$ docker rm <container id>
```

28. How to delete an image from the local storage system?

```
$ docker rmi <image-id>
```

29. How to build a Dockerfile?

Once you've written a Dockerfile, you need to build it to create an image with those specifications. Use the following command to build a Dockerfile:

```
$ docker build <path to docker file>
```

The next question would be when do you use “.dockerfile_name” and when to use the entire path?

Use “.dockerfile_name” when the dockerfile exists in the same file directory and you use the entire path if it lives somewhere else.

30. Do you know why *docker system prune* is used? What does it do?

\$ docker system prune

The above command is used to remove all the stopped containers, all the networks that are not used, all dangling images and all build caches. It's one of the most useful docker commands.

31. Will you lose your data, when a docker container exists?

No, you won't lose any data when Docker container exists. Any data that your application writes to the container gets preserved on the disk until you explicitly delete the container. The file system for the container persists even after the container halts.

32. Where all do you think Docker is being used?

When asked such a question, respond by talking about applications of Docker. Docker is being used in the following areas:

- Simplifying configuration: Docker lets you put your environment and configuration into code and deploy it.
- Code Pipeline Management: There are different systems used for development and production. As the code travels from development to testing to production, it goes through a difference in the environment. Docker helps in maintaining the code pipeline consistency.
- Developer Productivity: Using Docker for development gives us two things – We're closer to production and development environment is built faster.
- Application Isolation: As containers are applications wrapped together with all dependencies, your apps are isolated. They can work by themselves on any hardware that supports Docker.
- Debugging Capabilities: Docker supports various debugging tools that are not specific to containers but work well with containers.
- Multi-tenancy: Docker lets you have multi-tenant applications avoiding redundancy in your codes and deployments.
- Rapid Deployment: Docker eliminates the need to boost an entire OS from scratch, reducing the deployment time.

33. How is Docker different from other containerization methods?

Docker containers are very easy to deploy in any cloud platform. It can get more applications running on the same hardware when compared to other technologies, it makes it easy for developers to quickly create, ready-to-run containerized applications and it makes managing and deploying applications much easier. You can even share containers with your applications.

If you have some more points to add you can do that but make sure the above explanation is there in your answer.

34. Can I use JSON instead of YAML for my compose file in Docker?

You can use JSON instead of YAML for your compose file, to use JSON file with compose, specify the JSON filename to use, for eg:

```
$ docker-compose -f docker-compose.json up
```

35. How have you used Docker in your previous position?

Explain how you have used Docker to help rapid deployment. Explain how you have scripted Docker and used it with other tools like Puppet, Chef or Jenkins. If you have no past practical experience in Docker and instead have experience with other tools in a similar space, be honest and explain the same. In this case, it makes sense if you can compare other tools to Docker in terms of functionality.

36. How far do Docker containers scale? Are there any requirements for the same?

Large web deployments like Google and Twitter and platform providers such as Heroku and dotCloud, all run on container technology. Containers can be scaled to hundreds of thousands or even millions of them running in parallel. Talking about requirements, containers require the memory and the OS at all the times and a way to use this memory efficiently when scaled.

37. What platforms does docker run on?

This is a very straightforward question but can get tricky. Do some company research before going for the interview and find out how the company is using Docker. Make sure you mention the platform company is using in this answer.

Docker runs on various Linux administration:

- Ubuntu 12.04, 13.04 et al
- Fedora 19/20+
- RHEL 6.5+
- CentOS 6+
- Gentoo
- ArchLinux
- openSUSE 12.3+
- CRUX 3.0+

It can also be used in production with Cloud platforms with the following services:

- Amazon EC2
- Amazon ECS
- Google Compute Engine
- Microsoft Azure
- Rackspace

38. Is there a way to identify the status of a Docker container?

There are six possible states a container can be at any given point – Created, Running, Paused, Restarting, Exited, Dead.

Use the following command to check for docker state at any given point:

```
$ docker ps
```

The above command lists down only running containers by default. To look for all containers, use the following command:

```
$ docker ps -a
```

39. Can you remove a paused container from Docker?

The answer is no. You cannot remove a paused container. The container has to be in the stopped state before it can be removed.

40. Can a container restart by itself?

No, it's not possible for a container to restart by itself. By default the flag `-restart` is set to false.

41. Is it better to directly remove the container using the `rm` command or stop the container followed by remove container?

It's always better to stop the container and then remove it using the remove command.

```
$ docker stop <container_id>
```

```
$ docker rm -f <container_id>
```

Stopping the container and then removing it will allow sending `SIG_HUP` signal to recipients. This will ensure that all the containers have enough time to clean up their tasks. This method is considered a good practice, avoiding unwanted errors.

42. Will cloud overtake the use of Containerization?

Docker containers are gaining popularity but at the same time, Cloud services are giving a good fight. In my personal opinion, Docker will never be replaced by Cloud. Using cloud services with containerization will definitely hype the game. Organizations need to take their requirements and dependencies into consideration into the picture and decide what's best for them. Most of the companies have integrated Docker with the cloud. This way they can make the best out of both the technologies.

43. How many containers can run per host?

There can be as many containers as you wish per host. Docker does not put any restrictions on it. But you need to consider every container needs storage space, CPU and memory which the hardware needs to support. You also need to consider the application size. Containers are considered to be lightweight but very dependant on the host OS.

44. Is it a good practice to run stateful applications on Docker?

The concept behind stateful applications is that they store their data onto the local file system. You need to decide to move the application to another machine, retrieving data becomes painful. I honestly would not prefer running stateful applications on Docker.

45. Suppose you have an application that has many dependant services. Will docker compose wait for the current container to be ready to move to the running of the next service?

The answer is yes. Docker compose always runs in the dependency order. These dependencies are specifications like `depends_on`, `links`, `volumes_from`, etc.

46. How will you monitor Docker in production?

Docker provides functionalities like docker stats and docker events to monitor docker in production. Docker stats provides CPU and memory usage of the container. Docker events provide information about the activities taking place in the docker daemon.

47. Is it a good practice to run Docker compose in production?

Yes, using docker compose in production is the best practical application of docker compose. When you define applications with compose, you can use this compose definition in various production stages like CI, staging, testing, etc.

48. What changes are expected in your docker compose file while moving it to production?

These are the following changes you need make to your compose file before migrating your application to the production environment:

- Remove volume bindings, so the code stays inside the container and cannot be changed from outside the container.
- Binding to different ports on the host.
- Specify a restart policy
- Add extra services like log aggregator

49. Have you used Kubernetes? If you have, which one would you prefer amongst Docker and Kubernetes?

Be very honest in such questions. If you have used Kubernetes, talk about your experience with Kubernetes and Docker Swarm. Point out the key areas where you thought docker swarm was more efficient and vice versa.

50. Are you aware of load balancing across containers and hosts? How does it work?

While using docker service with multiple containers across different hosts, you come across the need to load balance the incoming traffic. Load balancing and HAProxy is basically used to balance the incoming traffic across different available(healthy) containers. If one container crashes, another container should automatically start running and the traffic should be re-routed to this new running container. Load balancing and HAProxy works around this concept.

Nagios Interview Questions

Q1. What is Nagios?

Nagios is one of the monitoring tools that is used for Continuous monitoring of systems, applications, services, and business processes etc. in a DevOps culture. In the event of a failure, Nagios can alert technical staff of the problem, allowing them to begin remediation processes before outages affects business processes, end-users, or customers. With Nagios you don't have to explain why an unseen infrastructure outage affect your organization's bottom line.

Nagios Features	
Feature	Description
Monitoring	Its powerful script APIs allow easy monitoring of in-house and custom applications, services, and systems
Visibility & Awareness	It provides a centralized view of the entire monitored IT infrastructure with detailed status information
Problem Remediation	Alert acknowledgments in Nagios, provide communication on known issues and problem response
Proactive Planning	Trending and capacity planning add-ons are there in Nagios to aware you about the aging infrastructure
Reporting	Availability reports ensure SLAs are being met & provide a record of alerts, notifications, and alert response
Customizable Code	Since it is an open source software you get the full access to its source code
Large Community	Nagios is backed up by a community of more than 1 million+ users worldwide which provides free support

Few of its important features are:

Now, once you have defined what is Nagios, you can mention the various things that you can achieve using Nagios.

By using Nagios you can:

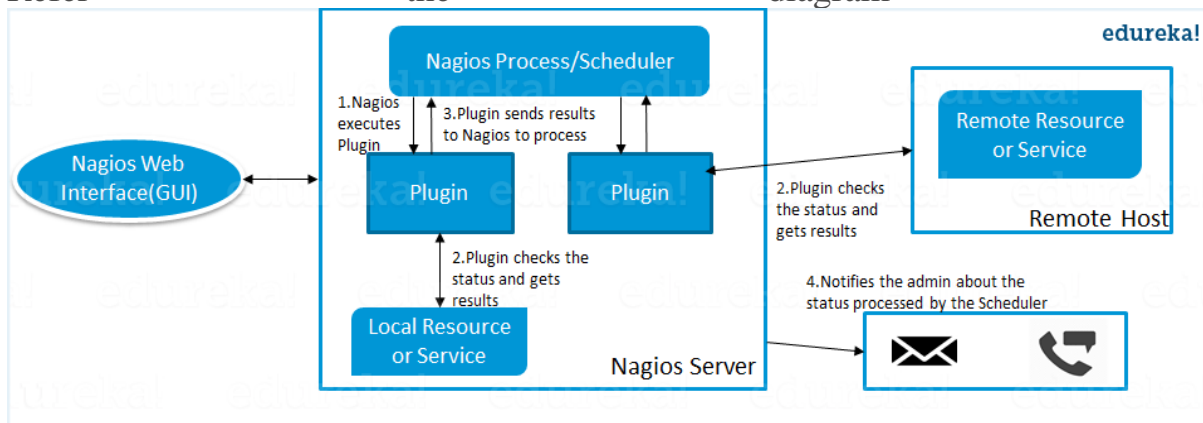
- Plan for infrastructure upgrades before outdated systems cause failures.
- Respond to issues at the first sign of a problem.
- Automatically fix problems when they are detected.
- Coordinate technical team responses.
- Ensure your organization's SLAs are being met.

- Ensure IT infrastructure outages have a minimal effect on your organization's bottom line.
- Monitor your entire infrastructure and business processes.

Q2. How does Nagios work?

- Nagios runs on a server, usually as a daemon or service.
- Nagios periodically runs plugins residing on the same server, they contact hosts or servers on your network or on the internet.
- One can view the status information using the web interface.
- You can also receive email or SMS notifications if something happens.
- The Nagios daemon behaves like a scheduler that runs certain scripts at certain moments.
- It stores the results of those scripts and will run other scripts if these results change.

Refer the diagram below:



Now, the next set of Nagios interview questions will focus on Nagios components like Plugins, NRPE, etc.

Q3. What are Plugins in Nagios?

Plugins are scripts (Perl scripts, Shell scripts, etc.) that can run from a command line to check the status of a host or service. Nagios uses the results from the plugins to determine the current status of hosts and services on your network.

Nagios will execute a Plugin whenever there is a need to check the status of a host or service. The plugin will perform the check and then simply returns the result to Nagios. Nagios will process the results that it receives from the Plugin and take the necessary actions.

Q4. What is NRPE (Nagios Remote Plugin Executor) in Nagios?

For this answer first give a small definition of NRPE.

The NRPE addon is designed to allow you to execute Nagios plugins on remote Linux/Unix machines. The main reason for doing this is to allow Nagios to monitor "local" resources (like CPU load, memory usage, etc.) on remote machines. Since these public resources are not

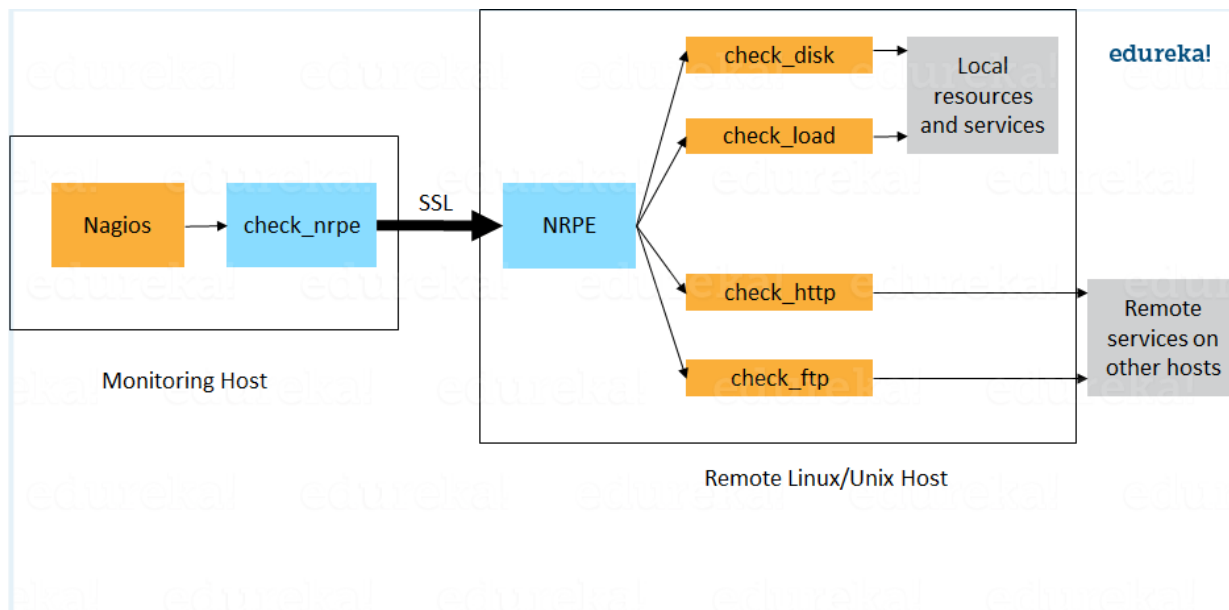
usually exposed to external machines, an agent like NRPE must be installed on the remote Linux/Unix machines.

The NRPE architecture on the basis of diagram shown below.

The NRPE addon consists of two pieces:

- The check_nrpe plugin, which resides on the local monitoring machine.
- The NRPE daemon, which runs on the remote Linux/Unix machine.

There is a SSL (Secure Socket Layer) connection between monitoring host and the remote host as shown in the diagram.



Q5. What is meant by Nagios backend?

Both Configuration and Logs can be stored in a backend. Configurations are stored in backend using NagiosQL. Historical data are stored using ndoutils. In addition, you also have nagdb and opdb.

Q6. What do you mean by passive check in Nagios?

Passive checks are initiated and performed by external applications/processes and the Passive check results are submitted to Nagios for processing.

Passive checks are useful for monitoring services that are Asynchronous in nature and cannot be monitored effectively by polling their status on a regularly scheduled basis. It can also be used for monitoring services that are Located behind a firewall and cannot be checked actively from the monitoring host.

Q7. When Does Nagios Check for external commands?

A) Make sure that you stick to the question during your explanation so I will advise you to follow the below mentioned flow:

Nagios check for external commands under the following conditions:

- At regular intervals specified by the `command_check_interval` option in the main configuration file or,
- Immediately after event handlers are executed. This is in addition to the regular cycle of external command checks and is done to provide immediate action if an event handler submits commands to Nagios.

Q8. What is the difference between Active and Passive check in Nagios?

The major difference between Active and Passive checks is that Active checks are initiated and performed by Nagios, while passive checks are performed by external applications.

Passive checks are useful for monitoring services that are:

- Asynchronous in nature and cannot be monitored effectively by polling their status on a regularly scheduled basis.
- Located behind a firewall and cannot be checked actively from the monitoring host.

The main features of Active checks are as follows:

- Active checks are initiated by the Nagios process.
- Active checks are run on a regularly scheduled basis.

Q9. How does Nagios help with Distributed Monitoring?

A)With Nagios you can monitor your whole enterprise by using a distributed monitoring scheme in which local slave instances of Nagios perform monitoring tasks and report the results back to a single master. You manage all configuration, notification, and reporting from the master, while the slaves do all the work. This design takes advantage of Nagios's ability to utilize passive checks i.e. external applications or processes that send results back to Nagios. In a distributed configuration, these external applications are other instances of Nagios.

Q10. Explain Main Configuration file of Nagios and its location?

A)The main configuration file contains a number of directives that affect how the Nagios daemon operates. This config file is read by both the Nagios daemon and the CGIs (It specifies the location of your main configuration file).

A sample main configuration file is created in the base directory of the Nagios distribution when you run the configure script. The default name of the main configuration file is `nagios.cfg`, it is usually placed in the `etc/` subdirectory of your Nagios installation (i.e. `/usr/local/nagios/etc/`).

Q11. Explain how Flap Detection works in Nagios?

Flapping occurs when a service or host changes state too frequently, this causes lot of problem and recovery notifications.

Whenever Nagios checks the status of a host or service, it will check to see if it has started or stopped flapping. Nagios follow the below procedure to do that:

- Storing the results of the last 21 checks of the host or service analyzing the historical check results and determine where state changes/transitions occur.
- Using the state transitions to determine a percent state change value (a measure of change) for the host or service.
- Comparing the percent state change value against low and high flapping thresholds
- A host or service is determined to have started flapping when its percent state change first exceeds a high flapping threshold.
- A host or service is determined to have stopped flapping when its percent state goes below a low flapping threshold.

Q12. What are the three main variables that affect recursion and inheritance in Nagios?

- Name
- Use
- Register

Name is a placeholder that is used by other objects. Use defines the “parent” object whose properties should be used. Register can have a value of 0 (indicating its only a template) and 1 (an actual object). The register value is never inherited.

Q13. What is meant by saying Nagios is Object Oriented?

One of the features of Nagios is object configuration format in that you can create object definitions that inherit properties from other object definitions and hence the name. This simplifies and clarifies relationships between various components.

Q14. What is State Stalking in Nagios?

State Stalking is used for logging purposes. When Stalking is enabled for a particular host or service, Nagios will watch that host or service very carefully and log any changes it sees in the output of check results.

It can be very helpful in later analysis of the log files. Under normal circumstances, the result of a host or service check is only logged if the host or service has changed state since it was last checked.

Q15. Nagios says my machine is unreachable, not down. What is the difference and how it is achieved?

When Nagios says a node is unreachable, a node is unreachable if Nagios is not able to find a path to the node.

The node itself may be up but because Nagios is unable to connect to it, it has to mark this as unreachable. To achieve this, Nagios use parent-child relationship between components.

A router may be defined as a parent for a server.

- Now Nagios checks for server and marks it as down.
- It then checks the parent (in our example, the router)
- If parent is also down, then server is marked as unreachable.
- If Parent is up, the server is marked as really down.

Q16. Explain Nagios state types?

The current state of monitored services and hosts is determined by two components:

- The status of service or host i.e. OK, WARNING, UP, DOWN etc..
- The type of state the service or host is in.

There are two types of states SOFT states and HARD states.

Now explain what is Soft and Hard states:

- When a service or host check results are in a non-OK or non-UP state and the service check has not yet been rechecked the number of times specified by the max_check_attempts directives in the service or host definition. This is called Soft Error. When a service or a host recovers from Soft Error that is considered as Soft Recovery.
- When a service or host check results are in a non-OK or non-UP state and the service check has been rechecked the number of times specified by the max_check_attempts directives in the service or host definition. This is called Hard Error. When a service or a host recovers from Hard Error that is considered as Hard Recovery.

SonarQube interview questions

Q1) What is SonarQube?

A) Sonar is a web based code quality analysis tool for Maven based Java projects. It covers a wide area of code quality check points which include: Architecture & Design, Complexity, Duplications, Coding Rules, Potential Bugs, Unit Test etc.

Q2) What is the use of SonarQube ?

A) Sonar covers the 7 sections of code quality

- Architecture and Design
- Unit tests
- Duplicated code
- Potential bugs
- Complex code
- Coding standards
- Comments

Q3) What are the advantages of using SonarQube?

A)

- SonarQube is open source
- SonarQube supports for various languages like Java, C#
- SonarQube reports for duplicate code, unit testing, code coverage, code complexity historical
- We can integrate SonarQube with build tools like ant, gradle
- SonarQube has Eclipse plugin like Sonarlint
- SonarQube supports external plugins like plugin for ldap

Q4) What are Quality Profiles in SonarQube?

A) The Quality Profiles service is central to SonarQube, since it is where you define your requirements by defining sets of rules (ex: Methods should not have a Cognitive Complexity greater than 15).

Ideally, all projects will be measured with the same profile for any given language, but that's not always practical. For instance, you may find that:

The technological implementation differs from one application to another (for example, different coding rules may apply when building threaded or non-threaded Java applications). You want to ensure stronger requirements on some of your applications (internal frameworks for example).

Q5) What are Quality Gates in SonarQube?

A) A quality gate is the best way to enforce a quality policy in your organization. It's there to answer ONE question: can I deliver my project to production today or not? In order to answer this question, you define a set of Boolean conditions based on measure thresholds against which projects are measured. For example:

No new blocker issues Code coverage on new code greater than 80% Etc. Ideally, all projects will be verified against the same quality gate, but that's not always practical. For instance, you may find that: Technological implementation differs from one application to another (you might not require the same code coverage on new code for Web or Java applications). You want to ensure stronger requirements on some of your applications (internal frameworks for example). Etc.

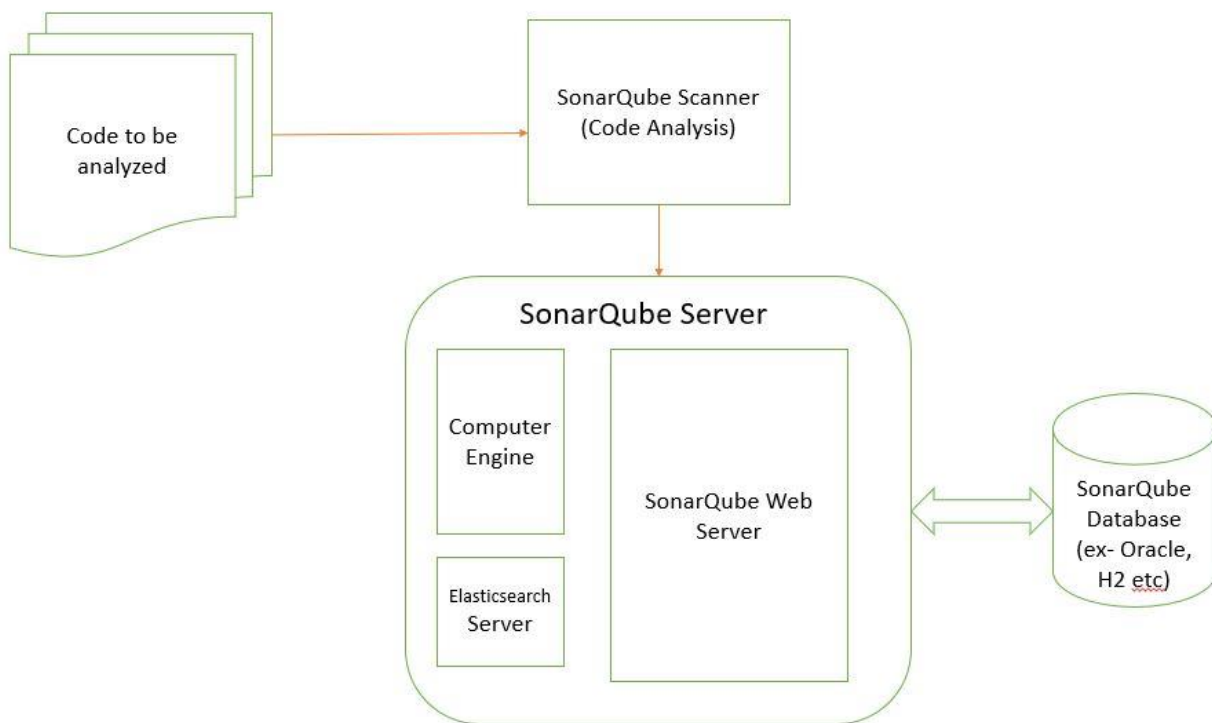
Which is why you can define as many quality gates as you wish. Quality Gates are defined and managed in the Quality Gates page found in the top menu.

Q6) What is role of database in SonarQube?

A) Sonar uses a Derby or H2 as default database. When running Sonar, it says that these databases may only be used for evaluation. We can change this default database and use our custom DB.

Q7) Explain architecture of SonarQube?

A)



Q8) How to create reports in SonarQube?

A) To create reports using SonarQube

mvn clean install

mvn sonar:sonar -D sonar.issuesreport.html.enable=true

Q9) What was the earlier name of SonarQube ?

A) Sonar

Q10) Where does SonarQube help ?

A) SonarQube provides report on coding standard, unit test, duplicate code, code coverage, potential bugs etc.

Q11) At which port sonar server is available by default ?

A)9000

Q12) Does SonarQube only analysis java code ?

A)No , SonarQube can analysis more than 20 languages.

Q13)In which language SonarQube is written ?

A)java

Q14) What are the main components of SonarQube Platform ?

- SonarQube plugin for languages
- SonarQube Scanner
- SonarQube Server
- SonarQube Database

Q15) What is the use of SonarQube Database ?

A)SonarQube Database stores configuration of the SonarQube instance like security settings and they also store project quality snapshot.

Q16)What is the use of SonarQube Scanners ?

A)It analyze projects on Continuous Integration Servers

Q17) Mention basic steps for SonarQube processing ?

- Developer develops code and sends its code into repository system like SCM, git
- An automatic build is fired in Continuous Integration Server and execution of SonarQube Scanner happens for SonarQube analysis.
- Report is sent to SonarQube Server for processing.
- SonarQube Server processes the report and stores the analysis report results in the SonarQube Database and displays the results in the UI
- Developers review, comment, challenge their Issues to manage and reduce their Technical Debt through the SonarQube UI.