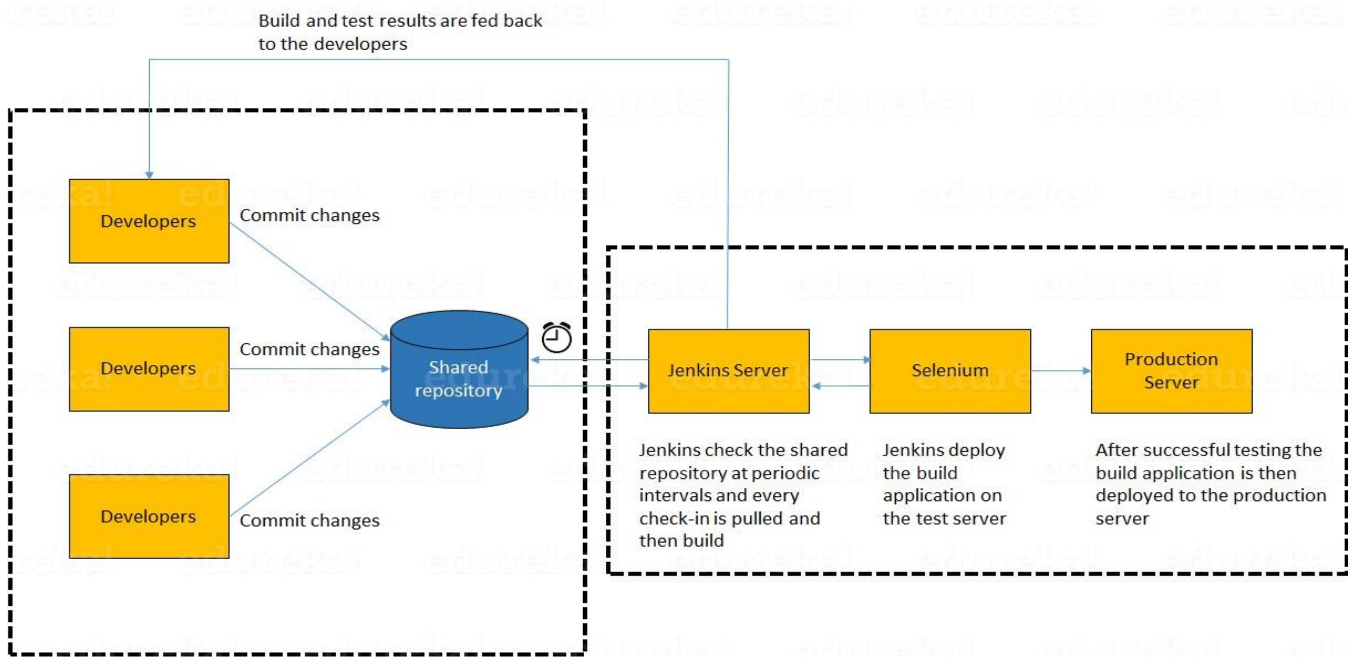
HUMMINGBIRD
Knowledge Hub

Continuous Integration, Continuous Testing, Configuration Management, Continuous Monitoring questions

Q1. What is meant by Continuous Integration?

I will advise you to begin this answer by giving a small definition of Continuous Integration (CI). It is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.
I suggest that you explain how you have implemented it in your previous job. You can refer the below given example:


Build and test results are fed back to the developers
Developers
Commit changes
Developers
Commit changes
Developers
Commit changes
Shared repository
Jenkins Server
Selenium
Production Server
Jenkins check the shared repository at periodic intervals and every check-in is pulled and then build
Jenkins deploy the build application on the test server
After successful testing the build application is then deployed to the production server

In the diagram shown above:

1. Developers check out code into their private workspaces.
2. When they are done with it they commit the changes to the shared repository (Version Control Repository).
3. The CI server monitors the repository and checks out changes when they occur.
4. The CI server then pulls these changes and builds the system and also runs unit and integration tests.
5. The CI server will now inform the team of the successful build.
6. If the build or tests fails, the CI server will alert the team.
7. The team will try to fix the issue at the earliest opportunity.
8. This process keeps on repeating.

Q2. Why do you need a Continuous Integration of Dev & Testing?

For this answer, you should focus on the need of Continuous Integration. My suggestion would be to mention the below explanation in your answer:

Continuous Integration of Dev and Testing improves the quality of software, and reduces the time taken to deliver it, by replacing the traditional practice of testing after completing all development. It allows Dev team to easily detect and locate problems early because developers need to integrate code into a shared repository several times a day (more frequently). Each check-in is then automatically tested.

Q3. What are the success factors for Continuous Integration?

Here you have to mention the requirements for Continuous Integration. You could include the following points in your answer:

- Maintain a code repository Automate the build
- Make the build self-testing
- Everyone commits to the baseline every day
- Every commit (to baseline) should be built
- Keep the build fast
- Test in a clone of the production environment
- Make it easy to get the latest deliverables
- Everyone can see the results of the latest build
- Automate deployment

Q4. 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 them 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.

Q5. Explain how can create a backup and copy files in Jenkins?

Answer to this question is really direct. 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 backup 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.

Q6. Explain how you can setup Jenkins job?

My approach to this answer will be to first mention how to create Jenkins job. Go to Jenkins top page, select "New Job", then choose "Build a free-style software project". Then you can tell 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..

Q7. Mention some of the useful plugins in Jenkins.

Below, I have mentioned some important Plugins:

- Maven 2 project
- 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 them as well. But, make sure you first mention the above stated plugins and then add your own.

Q8. How will you secure Jenkins?

The way I secure Jenkins is mentioned below. If you have any other way of doing it, please mention it in the comments section below:

- Ensure global security is on.
- Ensure that Jenkins is integrated with my company's user directory with appropriate plugin.
- Ensure that 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 same.

HUMMINGBIRD
Knowledge Hub

Continuous Testing

Q1. What is Continuous Testing?

I will advise you to follow the below mentioned explanation: Continuous Testing is the process of executing automated tests as part of the software delivery pipeline to obtain immediate feedback on the business risks associated with in the latest build. In this way, each build is tested continuously, allowing Development teams to get fast feedback so that they can prevent those problems from progressing to the next stage of Software delivery life-cycle. This dramatically speeds up a developer's workflow as there's no need to manually rebuild the project and re-run all tests after making changes.

Q2. What is Automation Testing?

Automation testing or Test Automation is a process of automating the manual process to test the application/system under test. Automation testing involves use of separate testing tools which lets you create test scripts which can be executed repeatedly and doesn't require any manual intervention.

Q3. What are the benefits of Automation Testing?

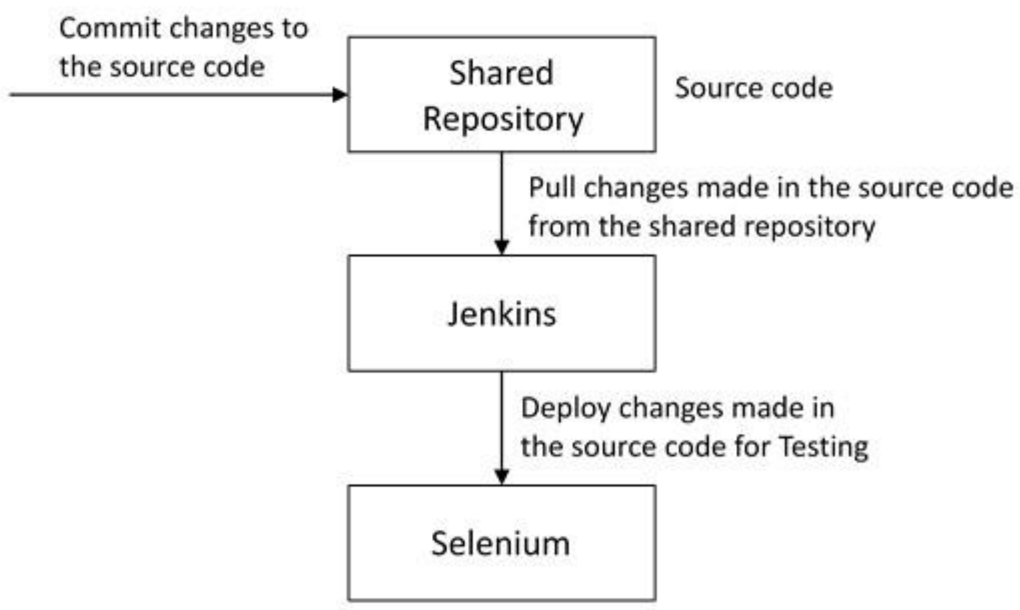
I have listed down some advantages of automation testing. Include these in your answer and you can add your own experience of how Continuous Testing helped your previous company:

- Supports execution of repeated test cases
- Aids in testing a large test matrix
- Enables parallel execution
- Encourages unattended execution
- Improves accuracy thereby reducing human generated errors
- Saves time and money

Q4. How to automate Testing in DevOps lifecycle?

I have mentioned a generic flow below which you can refer to: In DevOps, developers are required to commit all the changes made in the source code to a shared repository. Continuous Integration tools like Jenkins will pull the code from this shared repository every time a change is made in the code and deploy it for

Continuous Testing that is done by tools like Selenium as shown in the below diagram. In this way, any change in the code is continuously tested unlike the traditional approach.


Commit changes to the source code
Shared Repository
Source code
Pull changes made in the source code from the shared repository
Jenkins
Deploy changes made in the source code for Testing
Selenium

Q5. Why is Continuous Testing important for DevOps?

You can answer this question by saying, "Continuous Testing allows any change made in the code to be tested immediately. This avoids the problems created by having "bigbang" testing left to the end of the cycle such as release delays and quality issues. In this way, Continuous Testing facilitates more frequent and good quality releases." **Q6. What are the key elements of Continuous Testing tools?**

Key elements of Continuous Testing are:

- **Risk Assessment:** It Covers risk mitigation tasks, technical debt, quality assessment and test coverage optimization to ensure the build is ready to progress toward next stage.
- **Policy Analysis:** It ensures all processes align with the organization's evolving business and compliance demands are met.
- **Requirements Traceability:** It ensures true requirements are met and rework is not required. An object assessment is used to identify which requirements are at risk, working as expected or require further validation.
- **Advanced Analysis:** It uses automation in areas such as static code analysis, change impact analysis and scope assessment/prioritization to prevent defects in the first place and accomplishing more within each iteration.

- **Test Optimization:** It ensures tests yield accurate outcomes and provide actionable findings. Aspects include Test Data Management, Test Optimization Management and Test Maintenance
- **Service Virtualization:** It ensures access to real-world testing environments. Service visualization enables access to the virtual form of the required testing stages, cutting the waste time to test environment setup and availability.

Q7. Which Testing tool are you comfortable with and what are the benefits of that tool?

Here mention the testing tool that you have worked with and accordingly frame your answer. I have mentioned an example below:

I have worked on Selenium to ensure high quality and more frequent releases.

Some advantages of Selenium are:

- It is free and open source
- It has a large user base and helping communities
- It has cross Browser compatibility (Firefox, chrome, Internet Explorer, Safari etc.) It has great platform compatibility (Windows, Mac OS, Linux etc.)
- It supports multiple programming languages (Java, C#, Ruby, Python, Pearl etc.)
- It has fresh and regular repository developments
- It supports distributed testing

Q8. What are the Testing types supported by Selenium?

Selenium supports two types of testing:
Regression Testing: It is the act of retesting a product around an area where a bug was fixed.

Functional Testing: It refers to the testing of software features (functional points) individually.

Q9. What is Selenium IDE?

My suggestion is to start this answer by defining Selenium IDE. It is an integrated development environment for Selenium scripts. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests. Selenium IDE includes the entire Selenium Core, allowing you to easily and quickly record and play back tests in the actual environment that they will run in.

Now include some advantages in your answer. With autocomplete support and the ability to move commands around quickly, Selenium IDE is the ideal environment for creating Selenium tests no matter what style of tests you prefer.

Q10. What is the difference between Assert and Verify commands in Selenium?

I have mentioned differences between Assert and Verify commands below:

- Assert command checks whether the given condition is true or false. Let's say we assert whether the given element is present on the web page or not. If the condition is true, then the program control will execute the next test step. But, if the condition is false, the execution would stop and no further test would be executed.
- Verify command also checks whether the given condition is true or false. Irrespective of the condition being true or false, the program execution doesn't halts i.e. any failure during verification would not stop the execution and all the test steps would be executed.

Q11. How to launch Browser using WebDriver?

The following syntax can be used to launch Browser:

WebDriver driver = new FirefoxDriver();

WebDriver driver = new ChromeDriver();

WebDriver driver = new InternetExplorerDriver(); **Q12.**

When should I use Selenium Grid?

For this answer, my suggestion would be to give a small definition of Selenium Grid. It can be used to execute same or different test scripts on multiple platforms and browsers concurrently to achieve distributed test execution. This allows testing under different environments and saving execution time remarkably.

HUMMINGBIRD
Knowledge Hub

Configuration Management

Q1. What are the goals of Configuration management processes?

The purpose of Configuration Management (CM) is to ensure the integrity of a product or system throughout its life-cycle by making the development or deployment process controllable and repeatable, therefore creating a higher quality product or system. The CM process allows orderly management of system information and system changes for purposes such as to:

- Revise capability,
- Improve performance,
- Reliability or maintainability,
- Extend life,
- Reduce cost,
- Reduce risk and
- Liability, or correct defects.

Q2. What is the difference between Asset management and Configuration Management?

Given below are few differences between Asset Management and Configuration Management:

Asset Management	Configuration Management
Concerned with finances	Concerned with operations
Scope is everything you own	Scope is everything you deploy
Interfaces to purchasing and leasing	Interfaces to ITIL processes
Maintains data for taxes	Maintains data for troubleshooting
Lifecycle from Purchase to disposal	Lifecycle from deploy to retirement
Only incidental relationships	All operational relationships

Q3. What is the difference between an Asset and a Configuration Item?

According to me, you should first explain Asset. It has a financial value along with a depreciation rate attached to it. IT assets are just a sub-set of it. Anything and everything that has a cost and the organization uses it for its asset value calculation and related benefits in tax calculation falls under Asset Management, and such item is called an asset. Configuration Item on the other hand may or may not have financial values assigned to it. It will not have any depreciation linked to it. Thus, its life would not be dependent on its financial value but will depend on the time till that item becomes obsolete for the organization.

Now you can give an example that can showcase the similarity and differences between both:

1) Similarity:
Server – It is both an asset as well as a CI.
2) Difference :
Building – It is an asset but not a CI.
Document – It is a CI but not an asset

Q4. What do you understand by "Infrastructure as code"? How does it fit into the DevOps methodology? What purpose does it achieve?

Infrastructure as Code (IAC) is a type of IT infrastructure that operations teams can use to automatically manage and provision through code, rather than using a manual process. Companies for faster deployments treat infrastructure like software: as code that can be managed with the DevOps tools and processes. These tools let you make infrastructure changes more easily, rapidly, safely and reliably.

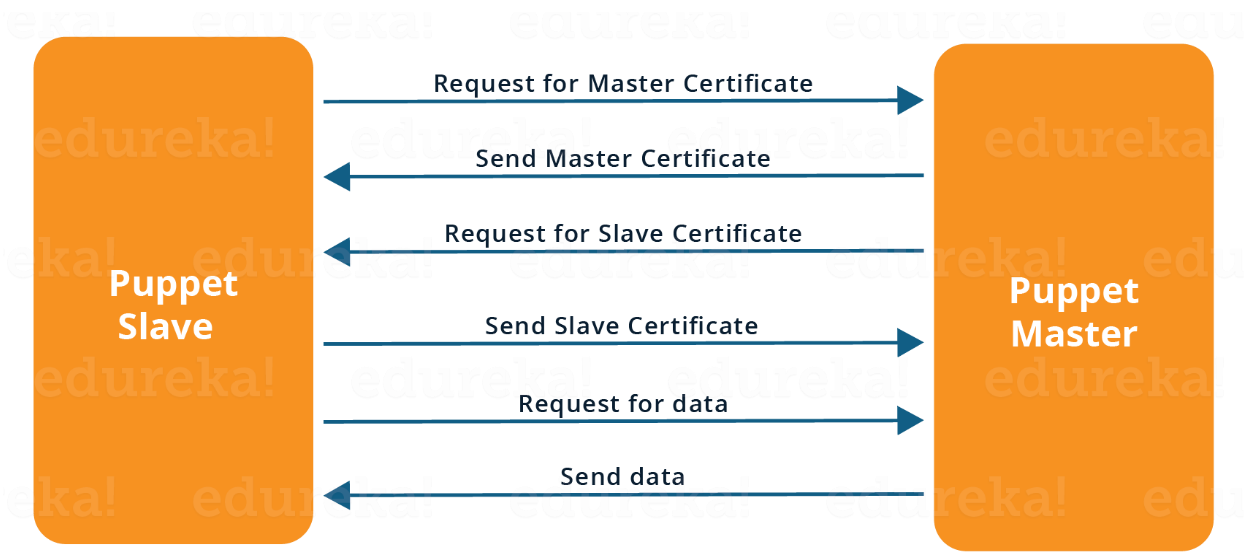
Q5. Which among Puppet, Chef, SaltStack and Ansible is the best Configuration Management (CM) tool? Why?

This depends on the organization's need so mention few points on all those tools:
Puppet is the oldest and most mature CM tool. Puppet is a Ruby-based Configuration Management tool, but while it has some free features, much of what makes Puppet great is only available in the paid version. Organizations that don't need a lot of extras will find

Puppet useful, but those needing more customization will probably need to upgrade to the paid version.
Chef is written in Ruby, so it can be customized by those who know the language. It also includes free features, plus it can be upgraded from open source to enterprise-level if necessary. On top of that, it's a very flexible product.
Ansible is a very secure option since it uses Secure Shell. It's a simple tool to use, but it does offer a number of other services in addition to configuration management. It's very easy to learn, so it's perfect for those who don't have a dedicated IT staff but still need a configuration management tool.
SaltStack is python based open source CM tool made for larger businesses, but its learning curve is fairly low.

Q6. What is Puppet?

I will advise you to first give a small definition of Puppet. It is a Configuration Management tool which is used to automate administration tasks.
Now you should describe its architecture and how Puppet manages its Agents. Puppet has a Master-Slave architecture in which the Slave has to first send a Certificate signing request to Master and Master has to sign that Certificate in order to establish a secure connection between Puppet Master and Puppet Slave as shown on the diagram below. Puppet Slave sends request to Puppet Master and Puppet Master then pushes configuration on Slave.
Refer the diagram below that explains the above description.



Q7. Before a client can authenticate with the Puppet Master, its certs need to be signed and accepted. How will you automate this task?

The easiest way is to enable auto-signing in puppet.conf. Do mention that this is a security risk. If you still want to do this:

- Firewall your puppet master – restrict port tcp/8140 to only networks that you trust.
- Create puppet masters for each 'trust zone', and only include the trusted nodes in that Puppet masters manifest.
- Never use a full wildcard such as *.

Q8. Describe the most significant gain you made from automating a process through Puppet.

For this answer, I will suggest you to explain you past experience with Puppet. you can refer the below example:

I automated the configuration and deployment of Linux and Windows machines using Puppet. In addition to shortening the processing time from one week to 10 minutes, I used the roles and profiles pattern and documented the purpose of each module in README to ensure that others could update the module using Git. The modules I wrote are still being used, but they've been improved by my teammates and members of the community

Q9. Which open source or community tools do you use to make Puppet more powerful?

Over here, you need to mention the tools and how you have used those tools to make Puppet more powerful. Below is one example for your reference:

Changes and requests are ticketed through Jira and we manage requests through an internal process. Then, we use Git and Puppet's Code Manager app to manage Puppet code in accordance with best practices. Additionally, we run all of our Puppet changes through our continuous integration pipeline in Jenkins using the beaker testing framework.

Q10. What are Puppet Manifests?

It is a very important question so make sure you go in a correct flow. According to me, you should first define Manifests. Every node (or Puppet Agent) has got its configuration details in Puppet Master, written in the native Puppet language. These details are written in the language which Puppet can understand and are termed as Manifests. They are composed of Puppet code and their filenames use the .pp extension.

Now give an exampl. You can write a manifest in Puppet Master that creates a file and installs apache on all Puppet Agents (Slaves) connected to the Puppet Master.

Q11. What is Puppet Module and How it is different from Puppet Manifest?

For this answer, you can go with the below mentioned explanation:
A Puppet Module is a collection of Manifests and data (such as facts, files, and templates), and they have a specific directory structure. Modules are useful for organizing your Puppet code, because they allow you to split your code into multiple Manifests. It is considered best practice to use Modules to organize almost all of your Puppet Manifests. Puppet programs are called Manifests which are composed of Puppet code and their file names use the .pp extension.

Q12. What is Facter in Puppet?

You are expected to answer what exactly Facter does in Puppet so according to me, you should say, "Facter gathers basic information (facts) about Puppet Agent such as hardware details, network settings, OS type and version, IP addresses, MAC addresses, SSH keys, and more. These facts are then made available in Puppet Master's Manifests as variables."

Q13. What is Chef?

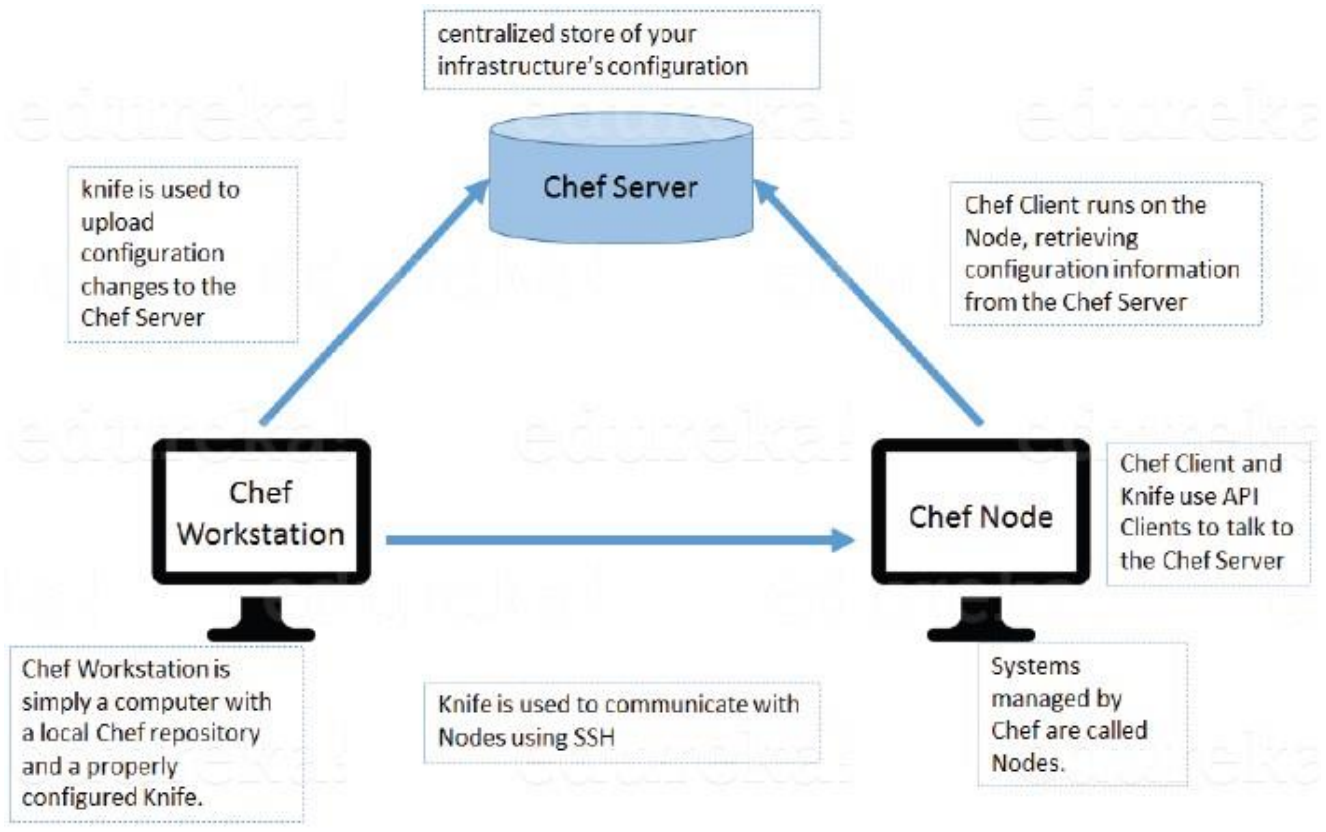
Begin this answer by defining Chef. It is a powerful automation platform that transforms infrastructure into code. Chef is a tool for which you write scripts that are used to automate processes. What processes? Pretty much anything related to IT.

Now you can explain the architecture of Chef, it consists of:

- **Chef Server:** The Chef Server is the central store of your infrastructure's configuration data. The Chef Server stores the data necessary to configure your nodes and provides search, a powerful tool that allows you to dynamically drive node configuration based on data.
- **Chef Node:** A Node is any host that is configured using Chef-client. Chef-client runs on your nodes, contacting the Chef Server for the information necessary to

configure the node. Since a Node is a machine that runs the Chef-client software, nodes are sometimes referred to as "clients".

- **Chef Workstation:** A Chef Workstation is the host you use to modify your cookbooks and other configuration data.


centralized store of your infrastructure's configuration
Chef Server
knife is used to upload configuration changes to the Chef Server
Chef Client runs on the Node, retrieving configuration information from the Chef Server
Chef Workstation
Chef Node
Chef Client and Knife use API Clients to talk to the Chef Server
Chef Workstation is simply a computer with a local Chef repository and a properly configured Knife.
Knife is used to communicate with Nodes using SSH
Systems managed by Chef are called Nodes.

Q14. What is a resource in Chef?

My suggestion is to first define Resource. A Resource represents a piece of infrastructure and its desired state, such as a package that should be installed, a service that should be running, or a file that should be generated.

You should explain about the functions of Resource for that include the following points:

- Describes the desired state for a configuration item.
- Declares the steps needed to bring that item to the desired state.
- Specifies a resource type such as package, template, or service.
- Lists additional details (also known as resource properties), as necessary.
- Are grouped into recipes, which describe working configurations.

Q15. What do you mean by recipe in Chef?

For this answer, I will suggest you to use the above mentioned flow: first define Recipe. A Recipe is a collection of Resources that describes a particular configuration or policy. A Recipe describes everything that is required to configure part of a system.

After the definition, explain the functions of Recipes by including the following points:

- Install and configure software components.
- Manage files.
- Deploy applications.
- Execute other recipes.

Q16. How does a Cookbook differ from a Recipe in Chef?

The answer to this is pretty direct. You can simply say, "a Recipe is a collection of Resources, and primarily configures a software package or some piece of infrastructure. A Cookbook groups together Recipes and other information in a way that is more manageable than having just Recipes alone."

Q17. What happens when you don't specify a Resource's action in Chef?

My suggestion is to first give a direct answer: when you don't specify a resource's action, Chef applies the default action.
Now explain this with an example, the below resource:

```
file 'C:\Users\Administrator\chef-repo\settings.ini' do
content 'greeting=hello world'
end
```

is same as the below resource:

```
file 'C:\Users\Administrator\chef-repo\settings.ini' do
action :create
content 'greeting=hello world'
end
```

because: create is the file Resource's default action.

Q18. What is Ansible module?

Modules are considered to be the units of work in Ansible. Each module is mostly standalone and can be written in a standard scripting language such as Python, Perl, Ruby, bash, etc.. One of the guiding properties of modules is idempotency, which means that even if an operation is repeated multiple times e.g. upon recovery from an outage, it will always place the system into the same state.

Q19. What are playbooks in Ansible?

Playbooks are Ansible's configuration, deployment, and orchestration language. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. Playbooks are designed to be human-readable and are developed in a basic text language.
At a basic level, playbooks can be used to manage configurations of and deployments to remote machines.

Q20. How do I see a list of all of the ansible_ variables?

Ansible by default gathers "facts" about the machines under management, and these facts can be accessed in Playbooks and in templates. To see a list of all of the facts that are available about a machine, you can run the "setup" module as an ad-hoc action:
Ansible -m setup hostname
This will print out a dictionary of all of the facts that are available for that particular host.

Q21. How can I set deployment order for applications?

WebLogic Server 8.1 allows you to select the load order for applications. See the Application MBean Load Order attribute in Application. WebLogic Server deploys serverlevel resources (first JDBC and then JMS) before deploying applications. Applications are deployed in this order: connectors, then EJBs, then Web Applications. If the application is an EAR, the individual components are loaded in the order in which they are declared in the application.xml deployment descriptor.

Q22. Can I refresh static components of a deployed application without having to redeploy the entire application?

Yes, you can use weblogic.Deployer to specify a component and target a server, using the following syntax: java weblogic.Deployer -adminurl http://admin:7001 -name appname -targets server1,server2 -deploy jsps/*.jsp

Q23. How do I turn the auto-deployment feature off?

The auto-deployment feature checks the applications folder every three seconds to determine whether there are any new applications or any changes to existing applications and then dynamically deploys these changes.

The auto-deployment feature is enabled for servers that run in development mode. To disable auto-deployment feature, use one of the following methods to place servers in production mode:

- In the Administration Console, click the name of the domain in the left pane, then select the Production Mode checkbox in the right pane.
- At the command line, include the following argument when starting the domain's Administration Server: -Dweblogic.ProductionModeEnabled=true
- Production mode is set for all WebLogic Server instances in a given domain.

Q24. When should I use the external_stage option?

Set -external_stage using weblogic.Deployer if you want to stage the application yourself, and prefer to copy it to its target by your own means.

HUMMINGBIRD
Knowledge Hub

Continuous Monitoring

Q1. Why is Continuous monitoring necessary?

I will suggest you to go with the below mentioned flow: Continuous Monitoring allows timely identification of problems or weaknesses and quick corrective action that helps reduce expenses of an organization. Continuous monitoring provides solution that addresses three operational disciplines known as:

- continuous audit
- continuous controls monitoring
- continuous transaction inspection

Q2. What is Nagios?

You can answer this question by first mentioning that Nagios is one of the monitoring tools. It 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 affect 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.
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.

This completes the answer to this question. Further details like advantages etc. can be added as per the direction where the discussion is headed.

Q3. How does Nagios works?

I will advise you to follow the below explanation for this answer: 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.

Now expect a few questions on Nagios components like Plugins, NRPE etc..

Q4. What are Plugins in Nagios?

Begin this answer by defining Plugins. They 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 Plugins to determine the current status of hosts and services on your network.
Once you have defined Plugins, explain why we need Plugins. Nagios will execute a Plugin whenever there is a need to check the status of a host or service. 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.

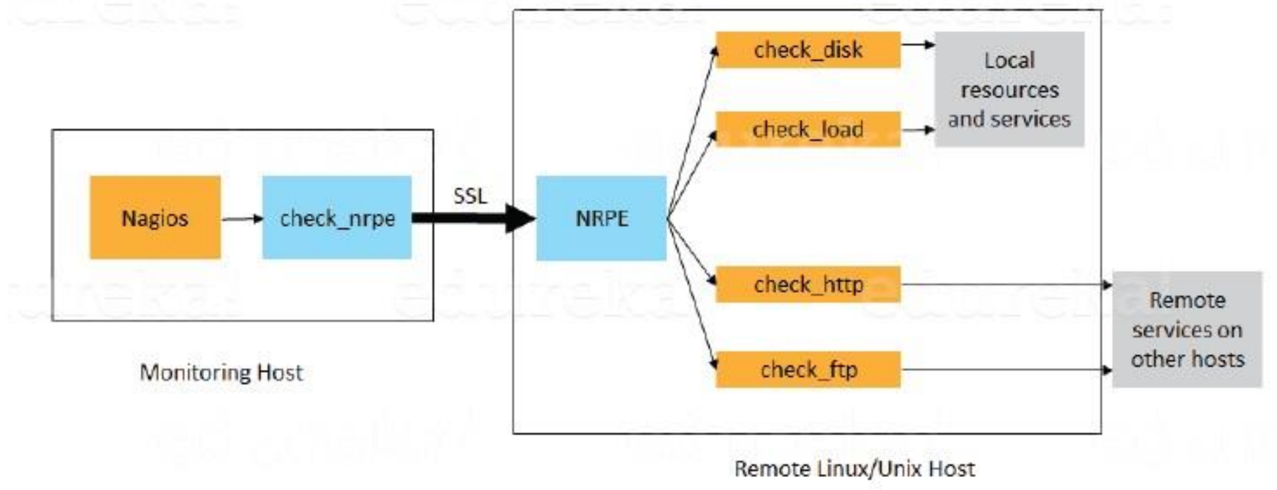
Q5. What is NRPE (Nagios Remote Plugin Executor) in Nagios?

For this answer, give a brief definition of Plugins. 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.

I will advise you to explain 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 remote host as shown in the diagram below.



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

According to me, the answer should start by explaining Passive checks. They are initiated and performed by external applications/processes and the Passive check results are submitted to Nagios for processing.

Then explain the need for passive checks. They are useful for monitoring services that are Asynchronous in nature and cannot be monitored effectively by polling their status on a regularly scheduled basis. They 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?

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?

For this answer, first point out the basic difference Active and Passive checks. 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. If your interviewer is looking unconvinced with the above explanation then you can also mention some key features of both Active and Passive checks:

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 Actives 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?

The interviewer will be expecting an answer related to the distributed architecture of Nagios. So, I suggest that you answer it in the below mentioned format:

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?

First mention what this main configuration file contains and its function. 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).

Now you can tell where it is present and how it is created. 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 you Nagios installation (i.e. /usr/local/nagios/etc/).

Q11. Explain how Flap Detection works in Nagios?

I will advise you to first explain Flapping first. Flapping occurs when a service or host changes state too frequently, this causes lot of problem and recovery notifications. Once you have defined Flapping, explain how Nagios detects Flapping. Whenever Nagios checks the status of a host or service, it will check to see if it has started or stopped flapping. Nagios follows the below given 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?

According to me the proper format for this answer should be: First name the variables and then a small explanation of each of these variables:

- Name
- Use
- Register

Then give a brief explanation for each of these variables. 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?

Answer to this question is pretty direct. I will answer this by saying, “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?

I will advise you to first give a small introduction on State Stalking. It 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. Depending on the discussion between you and interviewer you can also add, “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.”