This Jenkins interview questions and answers PDF for experienced DevOps professionals was originally published as an article on TheServerSide by Cameron McKenzie (@cameronmcnz). We highly recommend you visit TSS and view the original sample DevOps interview questions and answers article in order to view any changes or updates.

The Jenkins interview questions PDF

To be a full stack developer or a DevOps engineer, you need to know CI/CD. It is an absolute requirement. If you're applying for a new technical position and want to be prepared, here are 10 tough Jenkins interview questions and answers for DevOps engineers that employers often ask. By the way, this article is the second in a multi-part series of DevOps based interview questions. The first installment in the series can be found here: Tough sample DevOps interview questions and answers.

Jenkins interview questions and answers

A good strategy to use to apply to this set of tough Jenkins interview questions and answers for DevOps professionals is to first read through each question and formulate your own response. Then, read our answers. If you disagree, that is OK; just make sure you are ready to articulate your own ideas on the topic.

Jenkins interview questions PDF

Here are the 10 sample Jenkins interview questions for DevOps engineers. Before delving into the set of answers, read through this list, and see how comfortable you are coming up with your own responses to these tough Jenkins interview questions.

1. What are the software prerequisites that must be met before Jenkins is installed?
2. What is the syntax Jenkins uses to schedule items such as build jobs and SVN polling?
3. Name a Jenkins environment variable you have used in a shell script or batch file.
4. Name three security mechanisms Jenkins uses to authenticate users.
5. Describe the standard process to configure and use third-party tools within Jenkins.
6. Name two ways a Jenkins node agent can be configured to communicate back with the Jenkins master.
7. How do you take a backup of your Jenkins build jobs in order to prepare for disaster recovery?
8. Name three steps or stages a typical Jenkins pipeline might include.
9. How can you temporarily turn off Jenkins security if the administrative users have locked themselves out of the admin console?
10. Polling a Git repository for new commits is considered a Jenkins anti-pattern. What is a sound alternative to SVN polling?

Sample Jenkins interview questions

Here are the aforementioned answers to the advanced Jenkins interview.

1. What are the software prerequisites that must be met before Jenkins is installed?

Since version 2.54, Jenkins requires an installation of the Java Development Kit (JDK). JAVA_HOME should also be configured prior to installation. Version 1.8 of the JDK is the minimum that Jenkins will support.

Jenkins also requires the services of a Jakarta Enterprise Edition (EE) web profile-compliant servlet engine in order to run, although Jenkins comes with an embedded Jetty runtime that can be used if an existing Tomcat, WildFly or WebSphere application server is not available.

2. What is the syntax Jenkins uses to schedule items such as build jobs and SVN polling?

Jenkins uses the cron syntax to schedule various items within the tool.

The cron syntax is represented by five asterisks, with each one separated by a space. The first asterisk represents minutes, the second represents hours, the third the day of the month, the fourth the month itself and the fifth the day of the week. For example, to schedule a build job to pull from GitHub every Friday at 5:30 p.m., the syntax would be: `30 17 * * 4`.

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

There are a number of 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.

This isn't one of the most overly advanced Jenkins interview questions and answers, but it does demonstrate that the interviewee has indeed created a scripted Jenkins build job before.

4. Name three security mechanisms Jenkins uses to authenticate users.

Jenkins can authenticate users in one of three ways:

1. Jenkins can use an internal database to store user data and credentials. (This is the default.)
2. Jenkins can be configured to authenticate against a Lightweight Directory Access Protocol server.
3. Jenkins can be configured to employ the authentication mechanism used by the application server upon which it is deployed.

5. Describe the standard process to configure and use third-party tools within Jenkins?

The process to use a third-party tool, such as Artifactory, Node, SonarQube or Git typically follows a four-step process.

1. The third-party software must be installed.
2. A Jenkins plug-in that supports the third-party tool must be installed through the Jenkins admin console.
3. The third-party tool must be configured in the Tools tab of the Manage Jenkins section of the admin console.
4. Finally, the plug-in can be used from within a Jenkins build job. The plug-in will then facilitate communication between the Jenkins build job and the third-party tool.

This is a tough Jenkins interview question for DevOps professionals because not every third-party tool is configured in exactly the same way. For example, Jenkins can be configured to install Maven itself, rather than requiring a pre-existing installation. Similarly, third-party tools, like Checkstyle or JaCoCo, can be downloaded at build time by Maven. So these four steps are not always adhered to strictly, but at a high level, these are the typical steps required to install and configure a third-party Jenkins tool.

6. Name two ways a Jenkins node agent can be configured to communicate back with the Jenkins master.

The tool provides two mechanisms for starting a Jenkins node agent:

1. Launch a Jenkins node agent from a browser window.
2. Launch a Jenkins node agent from the command line.

When a Jenkins node agent is launched from a browser, a JNLP file is downloaded. When it runs, the JNLP file launches a new process on the client machine that runs Jenkins jobs.

To launch from the command line, the agent.jar file is required on the client. This executable JAR file is run from the command line, along with a reference to the slave agent's JNLP file that is hosted on the server. Like the JNLP file downloaded through a web browser, running this command launches a process on the client that can communicate with the Jenkins master and run Jenkins build jobs when it has idle clock cycles.

7. How do you take a backup of your Jenkins build jobs in order to prepare for disaster recovery?

Each Jenkins build stores its configuration as XML in a subdirectory of the JENKINS_HOME\jobs folder. By copying this folder to a secondary location, the configuration of all of the build jobs managed by the Jenkins master will be backed up as a result. Checking this folder into a source code management tool like Git isn't a bad idea either. Knowledge of ways to perform Jenkins Git integration is always looked upon fondly in a DevOps interview.

By simply copying the contents of this folder to a new Jenkins server instance, all of the build jobs described in this folder will be restored the next time the Jenkins server is started.

It's also worth noting that a history of all build jobs that have been run is written to this folder as well. To really impress the interviewer asking this advanced Jenkins interview question for DevOps, mention that using a .gitignore file and ignoring job-related files will ensure that any Git repository that stores Jenkins project configuration data will make sure only configuration data is stored. Build job history should be archived elsewhere and not stored in a source code repository.

8. Name three steps or stages a typical Jenkins pipeline might include.

A full-blown Jenkins pipeline will build a project from source code, put it through a variety of unit, integration, performance and user acceptance tests, and then, finally, if every test succeeds, deploy a packaged application to an application server, Nexus repository or Docker container. So, three fundamental stages would be:

1. Build
2. Test
3. Deploy

Of course, the best way to implement a Jenkins pipeline as code is to employ many modular steps. To really impress the interviewer asking one of these tough Jenkins interview questions for DevOps engineers, expand on these three stages, and describe a more complete Jenkins pipeline as code example.

Basic Jenkins interview questions and answers

A more comprehensive Jenkins pipeline might be broken down even further into something like this:

- Pull from GitHub using the Jenkins Git plug-in.
- Compile the application using the Jenkins Maven plug-in.
- Ensure developers are complying with coding standards by using the Jenkins Checkstyle Plugin.
- Calculate McCabe cyclomatic complexity, and using that data, calculate unit test coverage. The JaCoCo plug-in is often used to perform code coverage calculations.
- Check for bugs and potential security flaws with static code analysis tools, such as SonarQube, PMD or FindBugs.
- Obtain manual sign-off by the user acceptance team by including a Groovy script in the Jenkins pipeline.
- Run stress tests, and measure the application's performance under load.
- Package the application in a format suitable for deployment. This is often a WAR file for web apps, an EAR file for enterprise Java apps or an executable JAR file for Docker-based microservices.
- Deploy the application to a Maven repository, such as Nexus or Artifactory.
- Archive all of the generated reports for future reference.

9. 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 security is enabled, this file contains an XML element named `useSecurity` that will be set to `true`. By changing this setting to `false`, security will be disabled the next time Jenkins is restarted.

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

When asked one of these advanced Jenkins interview questions on DevOps security, be sure to emphasize that disabling security should always be both a last resort and a temporary measure. Once any authentication issues are resolved, be sure to re-enable Jenkins security and reboot the CI server.

10. Polling a Git repository for new commits is considered a Jenkins anti-pattern. What is a sound alternative to SVN polling?

Constantly polling a source code management tool like Git or Subversion to check if a new commit has been issued is a waste of clock cycles and should be avoided.

A better approach is to reverse this process and have the source code tool trigger a Jenkins build when new commits happen. With GitHub or GitLab, it is relatively easy to configure a post-commit hook that runs every time a commit is successful. When provided with the URL of the Jenkins build, the post-commit hook can easily trigger a Jenkins build, eliminating the need to have Jenkins constantly poll the source code repository.

Jenkins interview questions mastered

These are indeed a set of tough Jenkins interview questions and answers for DevOps engineers and aspiring cloud-native practitioners to tackle. But if you are comfortable with the responses provided here and can comfortably put these responses into your own words, you'll have no problem impressing your future employer with your ability to respond to a variety of advanced Jenkins interview questions that cover a wide and disparate range of DevOps topics.