CJE Exam Study Guide

Key Concepts of CI / CD and Jenkins

Definitions

Continuous integration (CI) is a software development practice where members of a team integrate their work frequently, at least daily, leading to multiple integrations per day.

Continuous Delivery (CD) is a software development discipline where software is built so that it *can* be released to production at any time.

Continuous Deployment extends Continuous Delivery by automating the deployment process, so that code which has passed automated testing is automatically deployed to production.

Pipeline, when not referencing the project type by that name, refers to the entire end to end process of software production.

This can be referred to as a continuous delivery pipeline or CI/CD pipeline, or a Continuous Deployment pipeline.

Jobs

What are jobs?
Jobs are runnable tasks that are controlled or monitored by Jenkins

What are the different types of jobs?
Freestyle, Monitor external, Matrix (Multiconfiguration), and Maven

What is meant by job scope?
All of the items that are within the context of a job

Builds

What is a build in Jenkins?
A build is the result of a single execution of a project.

What are build steps, triggers, artifacts, and repositories?
A build step is a single task within a project.

A trigger is a criteria for starting a new pipeline run or build.
An artifact is an immutable file that is generated during a build or pipeline run.
A repository is a location that holds items that need to be retrieved, source code, artifacts, jenkinsfiles etc.

Build tools configuration

Build tools are the software that is responsible for the build portion of the pipeline. Examples are Maven, Ant, etc.
Configuration varies by tool but the basic process is:

- Start Jenkins and install appropriate plugin
- Perform local configuration steps
- Create a pipeline that uses that build tool
- Update the tools configuration via the tools configuration file (POM, .cfg, etc).

Source

What is a Source Code Manager (SCM) and how is it used?
An SCM is a piece of software that is used for tracking changes in code. Changes are timestamped, and include the identity of the person making the change. Changes can be tracked and rolled back as needed. Versions of the code can be compared, stored, and merged with other versions.

Cloud-based SCMs

Git-hub and Gitlab are two exaxmples.

Jenkins Changelogs

These track the changes between builds in the Jenkins server itself.

Incremental Updates vs Clean Checkout

Incremental includes the working repository and the master repository combined. This can leave untracked files in the local repository.
Clean checkout involves removing all files in the local repository, and then pulling from the remote repository.

Checking in code

Checking in code is the process of pushing code updates to a repository. Checking code in is

the same as committing code. As part of the CI methodology, commits should be made frequently. Commits should each have a descriptive message indicating what changes the commit includes.

Infrastructure-as-Code

This is the process of managing and provisioning resources via configuration files. It allows us to maintain machine configurations in source control, and lets us roll back or version those configurations. This is especially prevalent in virtualization and containers.

Branch and Merge Strategies

These are the methods of checking code in and out of source control.
Branches from the main source can be short-lived or long-lived, and may be feature-based or team-based.
Merging is done in such a way that a *source of truth* is determined. One repository is determined to be the source of truth and changes that conflict are resolved in favor of that source.

Testing

Benefits of testing with Jenkins

Jenkins provides automation and notification for the testing process as well an insight via staging into the testing pipeline.
It makes testing an integral part of the build process by design.

Definitions

- Unit test: In this type, individual components (units) of software, are tested.
- Smoke test: This is a test of major functionality, for determining if the software is stable enough for a full test.
- Acceptance test: This tests functionality of the software, ensuring that it meets the guidelines set forth in the specifications. This is normally done by the *customer*.
- Automated Verification / Functional testing: In this type of testing, the end result is tested to determine if we "built what we set out to build," and answers the "Did we build it correctly" question.

Notifications

Types of Notifications

Email, SMS, and instant messaging are the three types of notifications, and they can be expanded via plugins.

Importance of Notifications: This is critical to the CI process. If a build fails, notification allows the dev team to correct the problem quickly. Also, in the event that a build needs manual approval to go to the deployment phase, this notifies the approver of a build being ready.

Distributed Builds

What are Distributed Builds?
They are build jobs in which the executor of the build is located on an agent (node) separate from the master.

The Functions of Masters and Agents: The master is the *Control* for the build farm. It collects reports and artifacts from the agents running on the nodes.
The agent software running on the node is then responsible for executing the portion of the pipeline that it has been assigned by the master.

Plugins

What are Plugins?
Plugins are extensions to Jenkins that add to its functionality.

What is the Plugin Manager?
This is the tool that is used to install, remove and update plugins. This tool can also be used to sort plugins and determine what is installed and what is available.

Jenkins Rest API

How do we interact with the API?
Interaction with the Jenkins API is done by invoking rest methods via the URL `.../api`, where `...` is the data that it acts on.

Why use the API?
The API can be used to programmatically create or copy jobs, retrieve information, and trigger jobs.

Security

Authentication vs Authorization
Authentication is verifying who you are.

Authorization is verifying what you can do.

Matrix Security
This is the ability to configure security based on different sections, or contexts, such as Credentials, Jobs, Nodes etc.
It is provided by one of the suggested plugins.
There is no explicit deny. Permission must be granted, and it propagates from parent to child unless the child is configured to not inherit.
If the child does not inherit permissions from its parent, permissions must be explicitly applied to the child in order to grant it permission.

Auditing
Auditing is the process of verifying that the access permissions are working as intended. As well as establishing a trail of user access in the system.

Credentials
A credential is any value that provides access to a restricted resource, this is also known as a secret.
Credential providers are the locations that have been configured for Jenkins to retrieve credentials

Fingerprints

What are fingerprints?
They are globally unique hashes that are used for tracking artifacts or other entities across multiple pipelines or projects.
Fingerprints are stored in the `$JENKINS_HOME/fingerprints/` directory. Fingerprints are stored under a hierarchy in there that is based on the first characters of the checksum.

How do fingerprints work?
Jenkins computes an MD5 hash for artifacts and entities that it fingerprints, and then it saves these with build data so that they can be mapped back to the build later. This allows artifacts to be mapped to the builds that generated them.

Artifacts

What is an artifact?
It is an immutable file that is the product of a project or pipeline run.

Why use artifacts?
They provide known good build code to end users in a repository.
They help prevent rebuilding unchanged parts of code.

Using them allows roll back and versioning of builds.

Storing Artifacts
Artifacts are stored in a repository, and fingerprinting is used to determine which builds produced them.
Retention policies can be configured to prevent bloating of the storage locations.
Storage locations can be defined, ensuring that certain file types are kept in known locations.
Artifacts can be shared between projects using the Copy Artifact plugin. This is not a suggested plugin and must be installed.

Using 3rd Party tools

3rd party tools include build tools and plugins that extend Jenkins functionality. These are configured and invoked from the pipeline either via the GUI or the jenkinsfile. Plugins contain information about the usage of the plugin on the link in the plugin manager.

Installation of Jenkins

What is the installation wizard?
The installation wizard is the graphical guided installer that is launched on port 8080 by default. It allows you to perform the initial configuration of Jenkins.

How is the installation wizard used?
First, unlock Jenkins by using the random `admin` password that is indicated on the Jenkins login page. Then proceed to selecting and installing plugins.

What plugins can be installed in the installation Wizard?
Suggested plugins and plugins included in the provided list.

Jobs

Organizing jobs in Jenkins is done using folders to logically group projects. Folders are also used to separate security contexts. The folder name becomes part of the job path and needs to be included. For example: `JENKINS_HOME/jobs/<foldername>/jobs/<buildname>`

Parameterized jobs are jobs that take an argument when they are run.
The parameter is a `key=value` that is passed as an environment variable to the job. In the case of the file parameter type, parameters can be used to upload files to the build.
The Parameterized Trigger Plugin is used to trigger a downstream job while passing a parameter.
There are different types of jobs based on the expected process, freestyle jobs being the most

common. Pipeline jobs invoke code as the job's configuration. Matrix jobs are multi-configuration jobs that combine the other job types.

Builds

Build steps are the portion of the project where work gets done.
Builds can be started by triggers, and can start other builds via triggers.
Builds can run scripts, either shell or batch, as a build step.
Build steps are where build tools are invoked.

Source Code Management

Source Code Management is also referred to as SCM.
SCM repositories are where code is stored.
Branches are sub sets of code in a repository that are used to group code logically, based on its state. *Master*, *testing*, and *development* are examples.
Polling refers to the scheduled checking of the SCM for changes.
Webhooks are notifications that are configured on the SCM which are sent to the Jenkins server. They go out when a change is made to the SCM repository that is being monitored by the webhook.
Webhooks remove the delay that is inherent in polling, as there is no need to wait for the scheduled polling to occur.
Tagging in the SCM can be used by Jenkins to differentiate code, such as version numbers and commit times.

Testing

This is performed by testing tools.
Once configured properly, Jenkins can consume the reports produced by these tools.
Jenkins provides native support for Junit test reports natively.
Code coverage testing refers to measuring the amount of code that is tested. More is better.
In the post-build actions section of the project, *Publish test reports* is an option that allows you to specify the location of the test reports.
The test reports will become available on the *Jobs* dashboard page.

Pipeline as Code

Value Stream Mapping enables visualizing the development process, identifying stages in the pipeline and seeing where inefficiency is occurring.

Pipelines in Jenkins allow you to define the build in code and then commit that code to SCM so that it can be versioned and tracked.
Pipeline code files are called Jenkinsfile(s).
A gate in a pipeline is a place where a manual step or a hand off occurs, such as manual approval for deployment of a build.
Keeping pipeline code in folders in jenkins allows you to lock down those pipelines to prevent cross team issues.
Pipelines can be broken into stages that are displayed on the project dashboard to allow insight into the steps of the process.
Credentials, if needed, can be passed in pipelines using the ID from the Jenkins repository.
Environemnt setup can be included as a script in the pipeline as a shell step if needed.

Triggering

Upstream jobs occur before other jobs.
Downstream jobs occur after other jobs.
Jobs can be configured to trigger other jobs.
Push is an active notification of another project. This is more immediate.
Pull is when a project watches another project for changes. This requires a polling interval to elapse.

Pipeline projects

Pipelines…

- Are superior to linked jobs
  - They are organized by stages and keep all of the related jobs in one space.
- Can be stored in jenkinsfiles and then committed to SCM
- Are resumable and can survive a master restart
- Can invoke complex functions and logic to process job steps
- Provide stage views on the dashboard so that project stages can be viewed and the status shown

Pipeline stages are named so that job status can be seen at a glance.
Pipeline Global libraries (shared libraries) allow code reuse between pipelines.
Shared libraries at the global level are "trusted" and can run any methods.
Trusted Libraries should be vetted and protected since they can run any methods.
Untrusted libraries are shared libraries that are not in the global scope.

- Untrusted libraries are monitored and are not allowed to run certain methods without admin approval.

- Code in an untrusted library runs in the "Groovy Sandbox," which keeps the code from affecting the node it is running on.

Multibranch Pipelines scan a repository for subbranches and will create a project for each branch when a jenkinsfile is located.

Promotion

Promoting builds are done to ensure that pretesting is available for specific builds.
Promoted builds are indicated by a configurable icon on the jobs dashboard.
Builds can be promoted based on conditionals, such as a list of downstream testing jobs that must succeed.
Job Promotion is an additional plugin that must be installed. It is not in the suggested plugins.
Artifacts from promoted builds can be ingested by other jobs using the Copy Artifact plugin.
This is also an add-on plugin.