Integrate Chef with a CI/CD pipeline for automated deployments

Table of Contents

Description

Integrating Chef with a CI/CD pipeline allows automated testing, configuration, and deployment of infrastructure changes. In this guide, you will configure a CI/CD pipeline using Chef to manage infrastructure and application deployment. This setup provides a consistent, testable approach to infrastructure provisioning and deployment.

Problem Statement

When managing infrastructure across environments, manual configurations can lead to inconsistencies. Automating deployment with Chef and integrating it into a CI/CD pipeline solves these issues by:

- Testing and validating configuration changes automatically.
- Applying configurations consistently across environments.
- Reducing human error in repetitive tasks.

Prerequisites

Software Required

- **Chef Workstation**: To write and test Chef configurations.
- **Chef Server**: For managing node configurations.
- **Docker**: For installing Jenkins.
- **Jenkins or another CI/CD tool**: To set up automated testing and deployment.
- **Git**: For version control and Chef cookbook management.

Hardware Requirement

- **Chef Server**: 4 GB RAM, 2 CPU cores minimum.
- **Workstation**: 4 GB RAM, 2 CPU cores minimum.
- **CI/CD Server** (e.g., Jenkins): 8 GB RAM, 4 CPU cores recommended.

Implementation Steps

Step 1: Install Jenkins Using Docker

1. Create Docker Network:

- Create a Docker network named "jenkins" to allow communication between containers.

```
docker network create jenkins
```

2. Run Docker Daemon Exposed with Port 2376:

- Start a Docker daemon with necessary configurations for Jenkins usage.

```
docker run --name jenkins-docker --rm --detach ^
--privileged --network jenkins --network-alias docker ^
--env DOCKER_TLS_CERTDIR=/certs ^
--volume jenkins-docker-certs:/certs/client ^
--volume jenkins-data:/var/jenkins_home ^
--publish 2376:2376 ^
docker:dind
```

3. Run Jenkins Image:

- Start Jenkins container using a custom image or the official Jenkins image, ensuring it's properly configured.

```
docker run --name jenkins-blueocean --restart=on-failure --detach ^
--network jenkins --env DOCKER_HOST=tcp://docker:2376 ^
--env DOCKER_CERT_PATH=/certs/client --env DOCKER_TLS_VERIFY=1 ^
--volume jenkins-data:/var/jenkins_home ^
--volume jenkins-docker-certs:/certs/client:ro ^
--publish 8080:8080 --publish 50000:50000 vijaynvb/jenkins:1.0
```

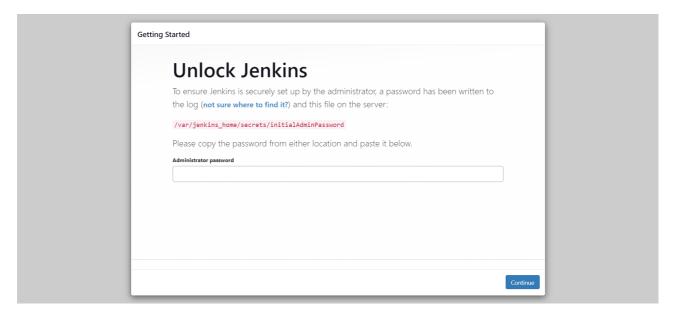- Open a browser and navigate to http://localhost:8080.

Setup admin details

- Access the jenkins application in localhost:8080

- Login with default credentials - To get the `password`, go to the shown directory or simply run `docker logs containerId` & get the password from logs.
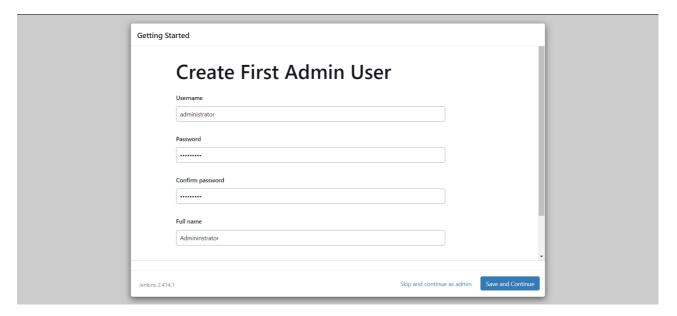

- copy the password from console paste it in login page.

- Select `Install suggested plugins`

- Create new user 'administrator' with password '******'

- Save and Continue

Step 2: Create a Windows Agent in Jenkins

1. **Go to Manage Jenkins**:

    ○ On the Jenkins dashboard, click **Manage Jenkins**.

＋ New Item

👥 People

🗄 Build History

◎ Project Relationship

🖐 Check File Fingerprint

⚙ Manage Jenkins

▢ My Views

🔵 Open Blue Ocean

2. **Go to Manage Nodes**:

  ○ Scroll down and click **Nodes** (or **Manage Nodes and Clouds**).

System Configuration

⚙ **System** Configure global settings and paths.	🔨 **Tools** Configure tools, their locations and automatic installers.	🧩 **Plugins** Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
🖥 **Nodes** Add, remove, control and monitor the various nodes that Jenkins runs jobs on.	☁ **Clouds** Add, remove, and configure cloud instances to provision agents on-demand.	

3. **Create New Node**:

  ○ Click **New Node** on the left sidebar.
  ○ Give the node a name (e.g., `windows-node`).
  ○ Select **Permanent Agent** and click **OK**.

New node

Node name

Type

○ **Permanent Agent**

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide
higher level of integration with these agents, such as dynamic provisioning. Select this type if no other
agent types apply — for example such as when you are adding a physical computer, virtual machines
managed outside Jenkins, etc.

○ **Copy Existing Node**

Create

4. **Configure the Windows Node**:

   ○ **Description**: Add an optional description.
   ○ **# Executors**: Set the number of executors (typically `1`).
   ○ **Remote Root Directory**: Set the directory where Jenkins stores the workspace (e.g.,
     `C:\jenkins`).
   ○ **Labels**: Add a label (e.g., `windows`) to easily reference the node in your pipeline.
   ○ **Launch Method**:
     ■ Select **Launch agent by connecting it to the controller

Remote root directory ?

```
C:\jenkins
```

Labels ?

```
windows
```

Usage ?

```
Use this node as much as possible                                              ⌄
```

Launch method ?

```
Launch agent by connecting it to the controller                                ⌄
```

5. **Save** the configuration.

Step 3: Start the Jenkins Agent on the Windows Node

1. Click on the **windows-node**

2. **Run the Jenkins Agent on Windows**:

   ○ Open **Command Prompt** (CMD) on the Windows node.
   ○ Copy the command from **Run from agent command line (Windows)** and run it in the windows **Command Prompt**.


```
curl.exe -sO http://172.19.4.192:8080/jnlpJars/agent.jar

java -jar agent.jar -jnlpUrl http://172.19.4.192:8080/computer/windows%2Dnode/jenkins-agent.jnlp -secret
68d850d43c092ea18c2d2de56dfb17719fcac291f4dfa16180f79ce24fb2d5fc -workDir "C:\jenkins"
```


3. **Verify Connection**:

   ○ After running the command, the Jenkins agent should connect to the Jenkins master.
   ○ The node will show as **online** on the **Manage Nodes** page in Jenkins.

Step 4: Create and Configure Credentials in Jenkins

4.1 Add Chef Server Credentials

- Go to **Manage Jenkins > Credentials > System > Global Credentials>Add Credentials**.

Security

Security
Secure Jenkins; define who is allowed to access/use the system.

Credentials
Configure credentials

Credential Providers
Configure the credential providers and types

Users
Create/delete/modify users that can log in to this Jenkins.

In-process Script Approval
Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions. 2 scripts pending approval.

Credentials

T	P	Store ↓	Domain
📱	🧑	System	(global)

System   + Add domain

Domain ↓	Description
🏰 Global credentials (unrestricted)	Credentials that should be available irrespective of domain specification to requirements matching.

Icon: S   M   L

Global credentials (unrestricted)   + Add Credentials

- Add a new credential:

  - **ID**: `chef-server-id`
  - **Username**: Chef Server username
  - **Password**: Chef Server password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)   ⌄

Username ?

<chef-server-username>

☐ Treat username as secret ?

Password ?

ID ?

<Chef-server-id>

Description ?

Create

Step 5: Add CHEF_WORKSTATION_PATH as a Global Environment Variable

5.1 Configure Global Properties

- Navigate to **Manage Jenkins > Configure System**.

System Configuration

⚙️ **System**
Configure global settings and paths.

🔨 **Tools**
Configure tools, their locations and automatic installers.

🧩 54 **Plugins**
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

🖥️ **Nodes**
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

☁️ **Clouds**
Add, remove, and configure cloud instances to provision agents on-demand.

- Enable **Global Properties > Environment Variables**.

- Add the following variable:

  - **Name**: CHEF_WORKSTATION_PATH
  - **Value**: C:/opscode/chef-workstation/bin

Global properties

☐ Disable deferred wipeout on this node ?
☑ Environment variables
    List of variables ?

    Name   ⊗

    CHEF_WORKSTATION_PATH

    Value

    C:\opscode\chef-workstation\bin

    Add

Step 6: Run the Pipeline

Once the agent is online, you can use the node in your pipeline as follows:

- Create a Pipeline

Enter an item name

Chef-Pipeline-1

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

- Add the pipeline struture as shown below.

- Click on **Apply** and **Save** the configurtion.
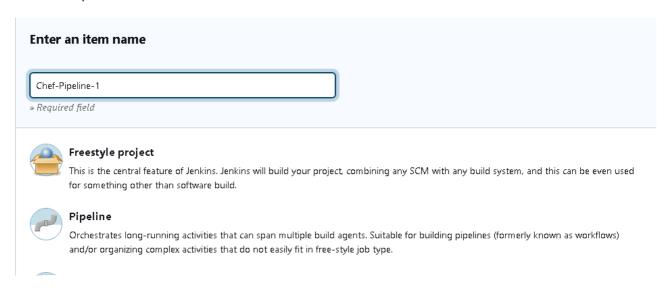
- Click on **Build Now** ro run the pipeline.

```
pipeline {
    agent { label 'windows' } // Ensure the pipeline runs on a Windows agent

    environment {
        CHEF_SERVER_CRED = credentials('sooryav') // Jenkins credential ID for
Chef server
        CHEF_ORG = 'chefserver02' // Chef organization name
        CHEF_NODE_IP = '172.19.3.101' // Target node IP (Ubuntu node)
        CHEF_NODE_USER = 'vagrant' // Node SSH username
        CHEF_NODE_PASS = 'vagrant' // Node SSH password
```

```
            CHEF_WORKSTATION_PATH = "C:/opscode/chef-workstation/bin" // Chef
Workstation path on Windows
            CHEF_LOCAL_PATH = "C:/Users/Administrator/Downloads/chef-starter/chef-
repo" // Local Chef folder path
            PATH = "${env.CHEF_WORKSTATION_PATH};${env.PATH}" // Add Chef Workstation
path to the PATH
    }

    stages {
        stage('Use Local Chef Repository') {
            steps {
                echo 'Using local Chef repository...'
                bat """
                echo Copying Chef repository from local path...
                xcopy /E /I /Y "${CHEF_LOCAL_PATH}" "%WORKSPACE%\\chef-repo"
                """
            }
        }

        stage('Upload Cookbooks') {
            steps {
                echo 'Uploading cookbooks to Chef server...'
                bat """
                %CHEF_WORKSTATION_PATH%/knife cookbook upload webserver --config
%WORKSPACE%\\chef-repo\\.chef\\knife.rb --cookbook-path %WORKSPACE%\\chef-
repo\\cookbooks
                """
            }
        }
    }

    post {
        success {
            echo 'Pipeline executed successfully! Chef configurations are applied
and verified.'
        }
        failure {
            echo 'Pipeline execution failed. Check the logs for detailed error
information.'
        }
    }
}
```

- Check the pipeline status on **Console output**.

✅ **Console Output**

```
Started by user admin
[Pipeline] Start of Pipeline
[Pipeline] node
Running on windows-node in C:\jenkins\workspace\Chef-Pipeline-1
[Pipeline] {
[Pipeline] withCredentials
Masking supported pattern matches of %CHEF_SERVER_CRED% or %CHEF_SERVER_CRED_PSW%
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Use Local Chef Repository)
[Pipeline] echo
Using local Chef repository...
[Pipeline] bat
```

- The pipeline begins by cloning the Chef repository from GitHub into the Jenkins workspace.

- Upload the `webserver` cookbook to the Chef server using `knife`.

Step-7: Update and Configure `knife.rb` for Jenkins Pipeline

To fix the `knife not found` error, you need to rename your `config.rb` file to `knife.rb` and ensure it is configured properly to work with the Jenkins pipeline structure. Below are the steps:

Steps to Configure `knife.rb`

1. **Locate the `config.rb` File**:

   ○ Navigate to the directory where `config.rb` is located. Typically, it will be in the `.chef` folder within your Chef repository.

   ```
   cd C:/Users/Administrator/Downloads/chef-starter/chef-repo/.chef
   ```

2. **Rename `config.rb` to `knife.rb`**:

   ○ Rename the `config.rb` file using the command:

   ```
   ren config.rb knife.rb
   ```

3. **Update `knife.rb` Configuration**:

   ○ Open the renamed `knife.rb` file in an editor (like Notepad++ or VS Code) and update it to work with your Jenkins pipeline structure:

```
# Knife configuration for Jenkins pipeline
current_dir = File.dirname(__FILE__)
log_level                   :info
log_location                STDOUT
node_name                   "sooryav"
client_key                  "#{ENV['WORKSPACE']}\\chef-
repo\\.chef\\sooryav.pem"
chef_server_url
"https://api.chef.io/organizations/chefserver02"
cookbook_path               ["#{ENV['WORKSPACE']}\\chef-repo\\cookbooks"]
```

References

1. **Chef Official Documentation**: https://docs.chef.io/
2. **Jenkins Official Documentation**: https://www.jenkins.io/
3. **Docker Documentation**: https://docs.docker.com/
4. **GitHub Repository Setup**: https://github.com/