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

Completion of all previous lab guides (up to Lab Guide-09) is required before proceeding with Lab Guide-10.

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
```

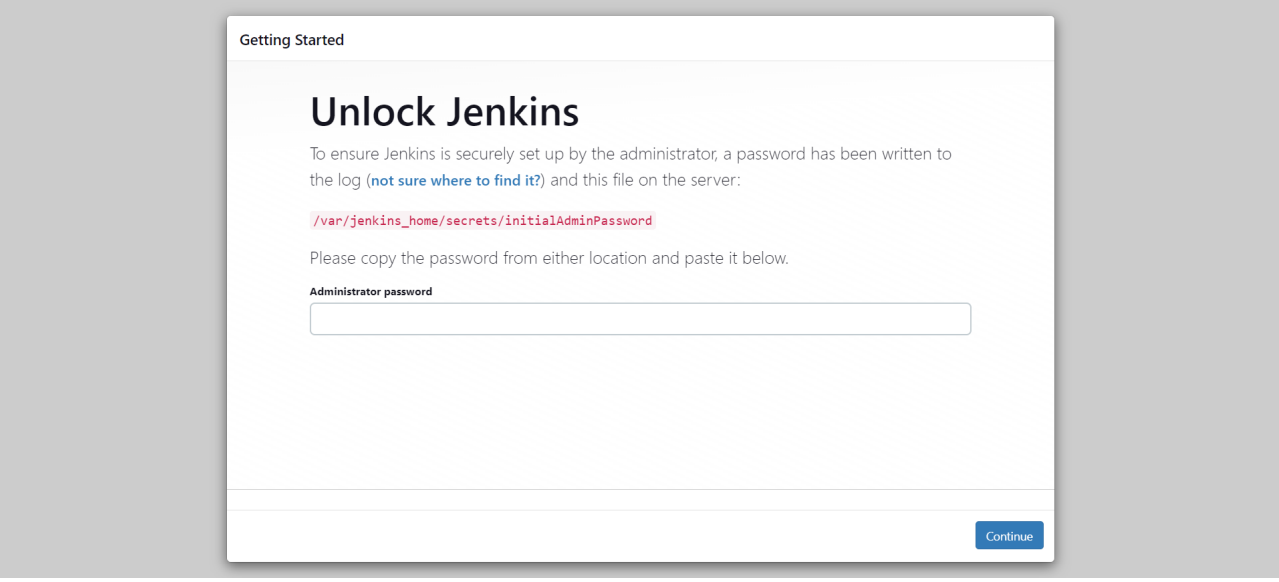
- Open a browser and navigate to `http://localhost:8080`.

Setup admin details

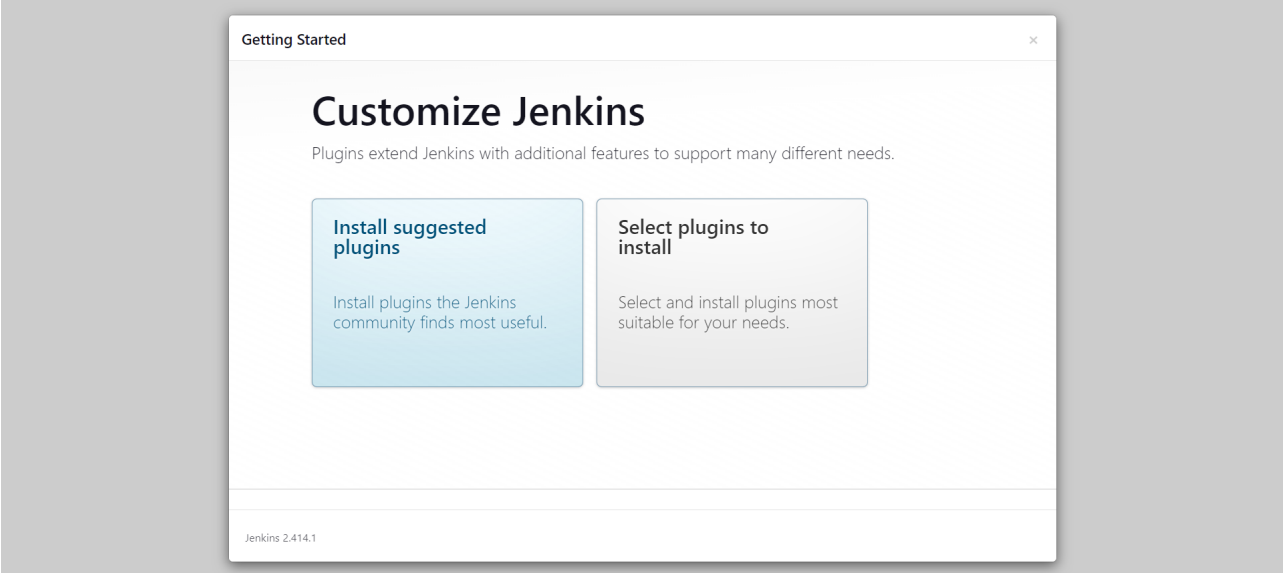
- Access the jenkins application in `localhost:8080`
- Login with default credentials - To get the `password` , go to the shown directory or simply run `docker logs containerId` & get the password from logs.

```
C:\Users\vijay>docker logs 01c76bd423c195725627c7688a316da3ecde547985eb7802920934e3b7c7bce5
Running from: /usr/share/jenkins/jenkins.war
webroot: /var/jenkins_home/war
2024-10-04 07:14:45.425+0000 [id=1]     INFO    winstone.Logger#logInternal: Beginning extraction from war file
2024-10-04 07:14:46.227+0000 [id=1]     WARNING o.e.j.s.handler.ContextHandler#setContextPath: Empty contextPath
2024-10-04 07:14:46.271+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: jetty-10.0.15; built: 2023-04-11T17:25:14.480Z; git: 68017dbd00236bb7e187330d7585a059610f661d; jvm
17.0.8+7
2024-10-04 07:14:46.466+0000 [id=1]     INFO    o.e.j.w.StandardDescriptorProcessor#visitServlet: NO JSP Support for /, did not find org.eclipse.jetty.jsp.JettyJspServlet
2024-10-04 07:14:46.506+0000 [id=1]     INFO    o.e.j.s.s.DefaultSessionIdManager#doStart: Session workerName=node0
2024-10-04 07:14:47.135+0000 [id=1]     INFO    hudson.WebAppMain#contextInitialized: Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnvVars.get("JENKINS_HOME")
2024-10-04 07:14:47.310+0000 [id=1]     INFO    o.e.j.s.handler.ContextHandler#doStart: Started w.@2c8662ac{Jenkins v2.414.1,/,file:///var/jenkins_home/war/,AVAILABLE}{/var/jenkins_home/wa
r}
2024-10-04 07:14:47.329+0000 [id=1]     INFO    o.e.j.server.AbstractConnector#doStart: Started ServerConnector@3569fc08{HTTP/1.1, (http/1.1)}{0.0.0.0:8080}
2024-10-04 07:14:47.344+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: Started Server@42f33b5d{STARTING}[10.0.15,sto=0] @2944ms
2024-10-04 07:14:47.346+0000 [id=27]    INFO    winstone.Logger#logInternal: Winstone Servlet Engine running: controlPort=disabled
2024-10-04 07:14:47.649+0000 [id=34]    INFO    jenkins.InitReactorRunner$1#onAttained: Started initialization
2024-10-04 07:14:48.264+0000 [id=32]    INFO    hudson.PluginManager#considerDetachedPlugin: Loading a detached plugin as a dependency: /var/jenkins_home/plugins/javax-mail-api.jpi
2024-10-04 07:14:49.665+0000 [id=32]    INFO    jenkins.InitReactorRunner$1#onAttained: Listed all plugins
2024-10-04 07:14:53.539+0000 [id=47]    INFO    jenkins.InitReactorRunner$1#onAttained: Prepared all plugins
2024-10-04 07:14:53.553+0000 [id=37]    INFO    jenkins.InitReactorRunner$1#onAttained: Started all plugins
2024-10-04 07:14:53.557+0000 [id=46]    INFO    jenkins.InitReactorRunner$1#onAttained: Augmented all extensions
2024-10-04 07:14:55.309+0000 [id=45]    INFO    jenkins.InitReactorRunner$1#onAttained: System config loaded
2024-10-04 07:14:55.310+0000 [id=32]    INFO    jenkins.InitReactorRunner$1#onAttained: System config adapted
2024-10-04 07:14:55.314+0000 [id=40]    INFO    jenkins.InitReactorRunner$1#onAttained: Loaded all jobs
2024-10-04 07:14:55.327+0000 [id=36]    INFO    jenkins.InitReactorRunner$1#onAttained: Configuration for all jobs updated
2024-10-04 07:14:55.350+0000 [id=61]    INFO    hudson.util.Retrier#start: Attempt #1 to do the action check updates server
2024-10-04 07:14:55.753+0000 [id=33]    INFO    jenkins.install.SetupWizard#init:

*************************************************************
*************************************************************
*************************************************************

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

779a25b990c3486e98b5250d1abe4a76

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*************************************************************
*************************************************************
*************************************************************

2024-10-04 07:15:15.740+0000 [id=33]    INFO    jenkins.InitReactorRunner$1#onAttained: Completed initialization
2024-10-04 07:15:15.775+0000 [id=26]    INFO    hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
2024-10-04 07:15:16.692+0000 [id=61]    INFO    h.m.DownloadService$Downloadable#load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
2024-10-04 07:15:16.693+0000 [id=61]    INFO    hudson.util.Retrier#start: Performed the action check updates server successfully at the attempt #1
```

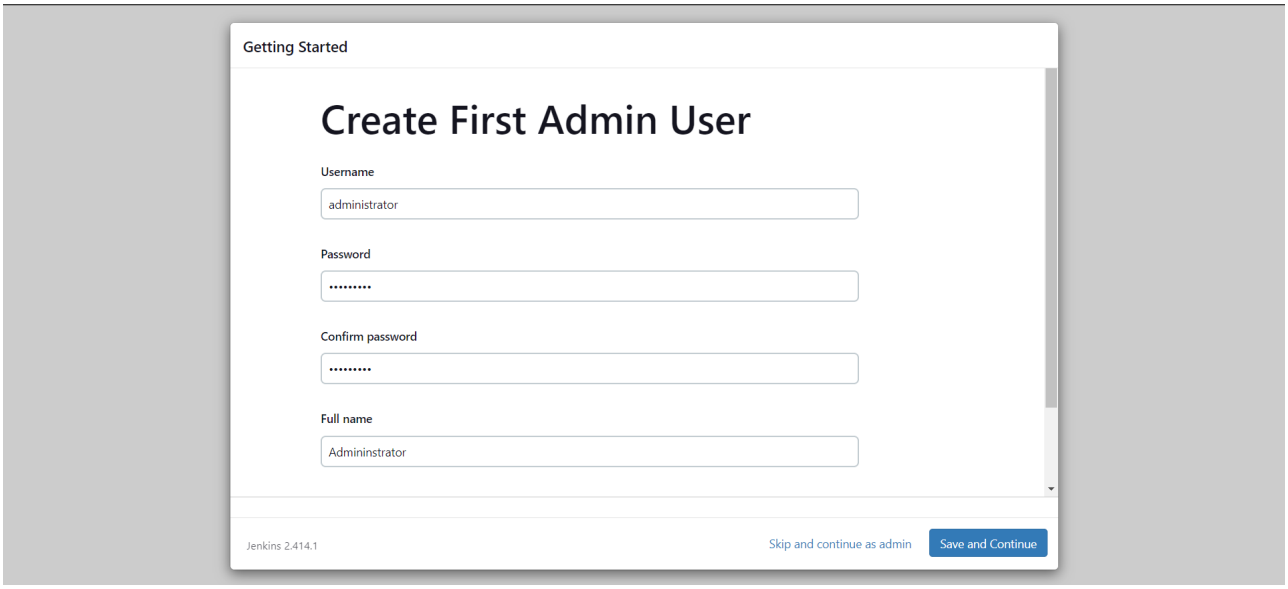
- copy the password from console paste it in login page.



- Select `Install suggested plugins`



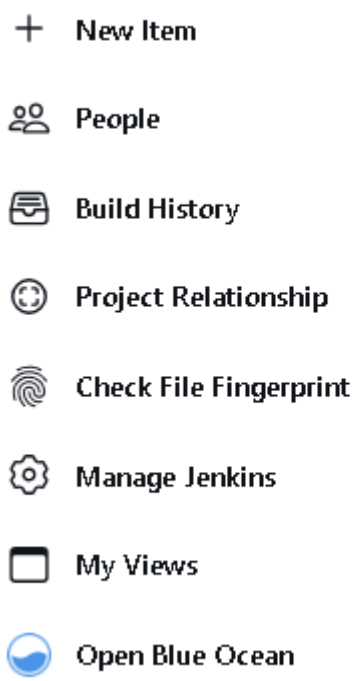
- Create new user 'administrator' with password '****'



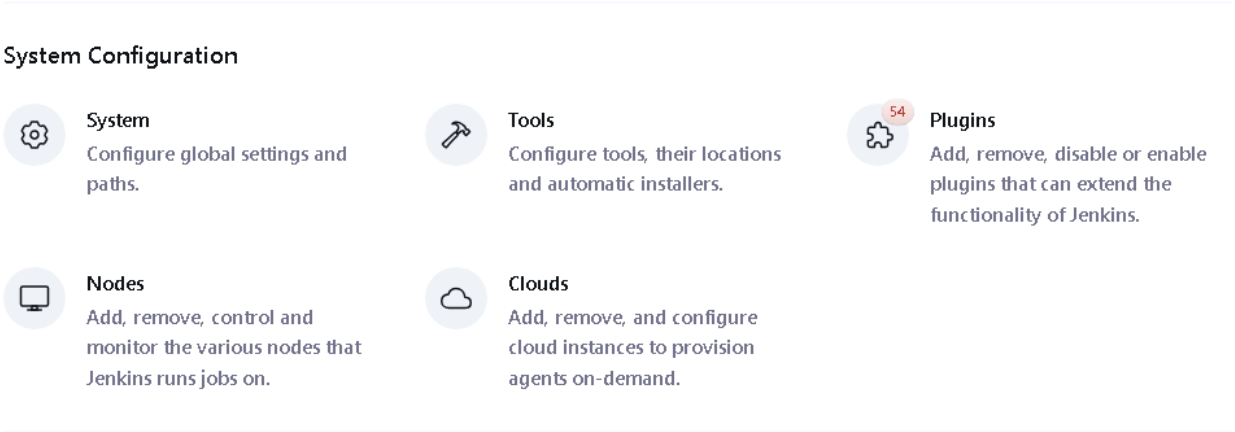
- Save and Continue

Step 2: Create a Windows Agent in Jenkins

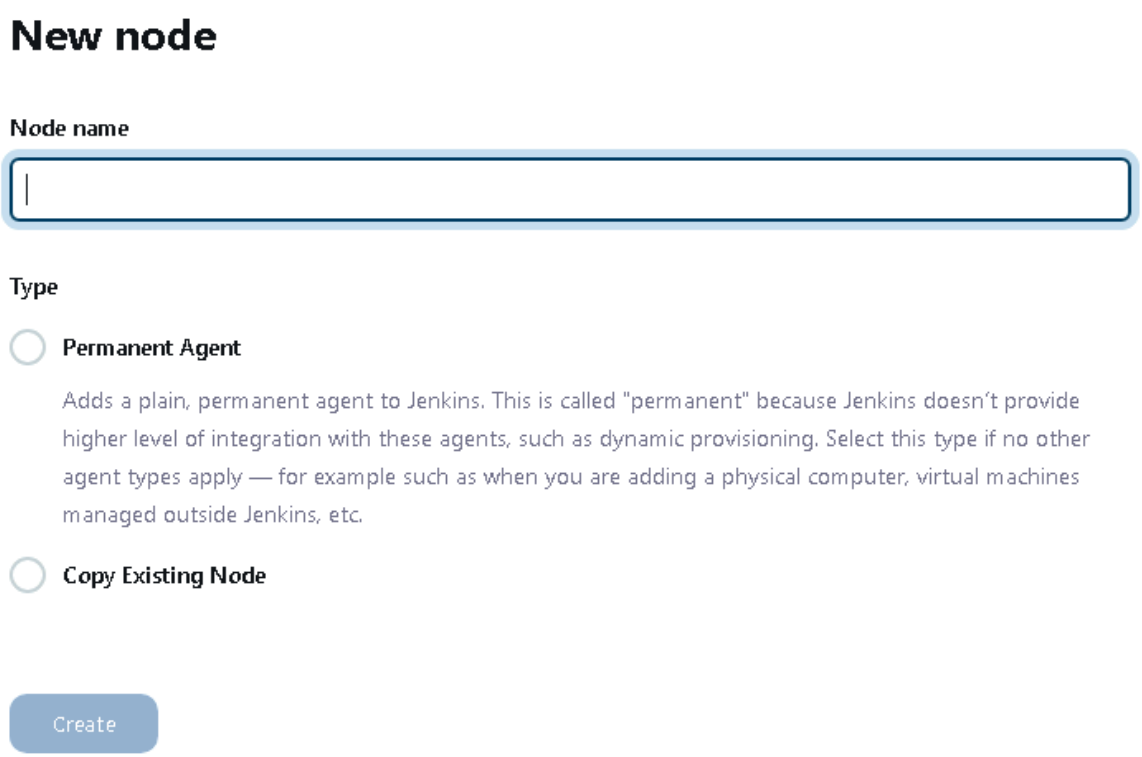
1. **Go to Manage Jenkins**:
    - On the Jenkins dashboard, click **Manage Jenkins**.



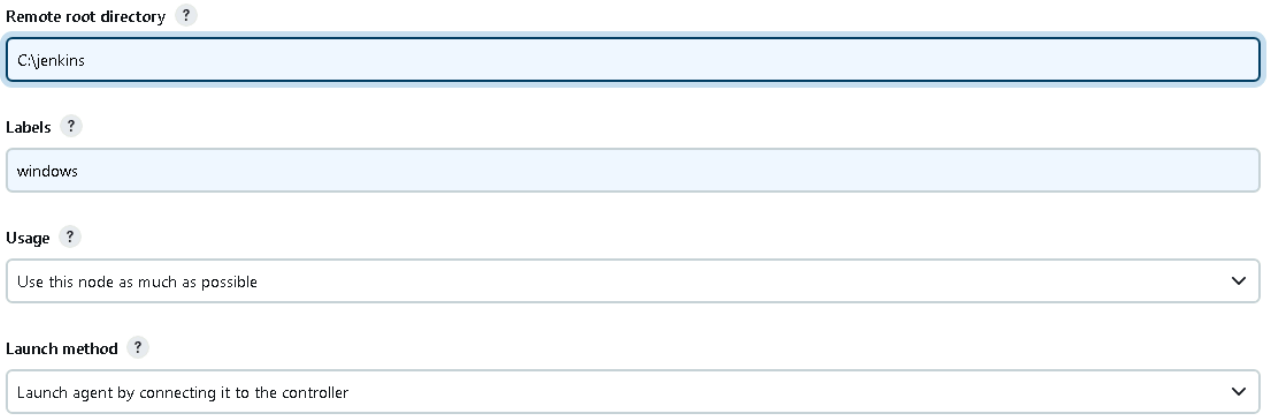
2. **Go to Manage Nodes**:
    - Scroll down and click **Nodes** (or **Manage Nodes and Clouds**).



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**.



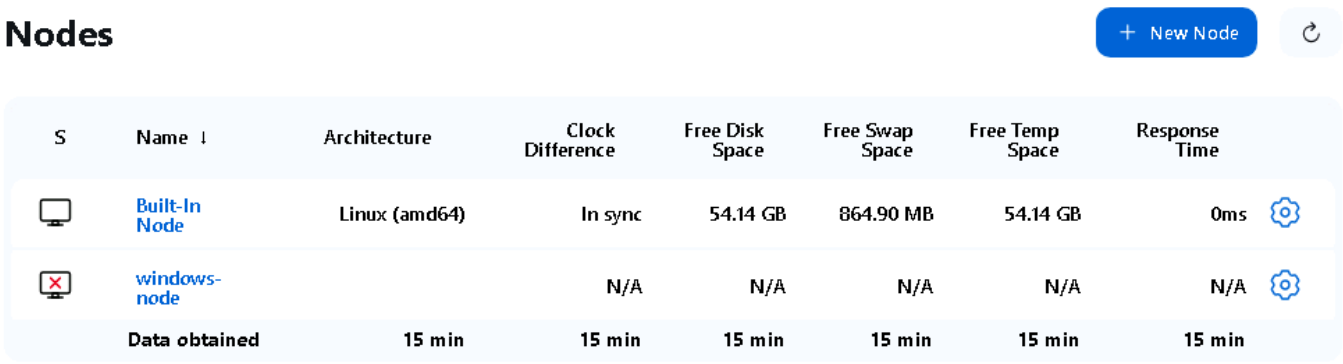
4. **Configure the Windows Node**:
    - **Description**: Add an optional description.
    - **# Executors**: Set the number of executors (typically `1`).
    - **Remote Root Directory**: Set the directory where Jenkins stores the workspace (e.g., `C:\jenkins`).
    - **Labels**: Add a label (e.g., `windows`)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



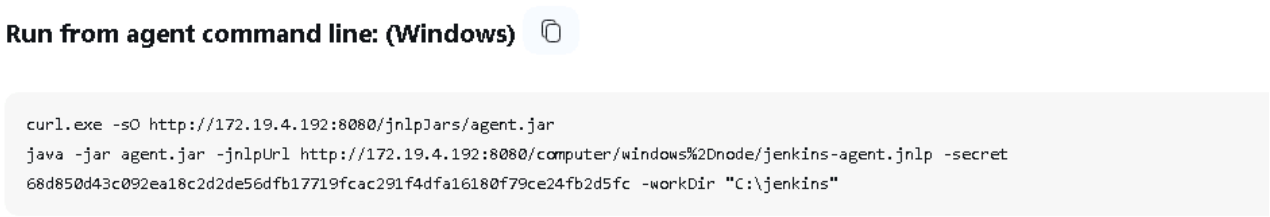
5. **Save** the configuration.

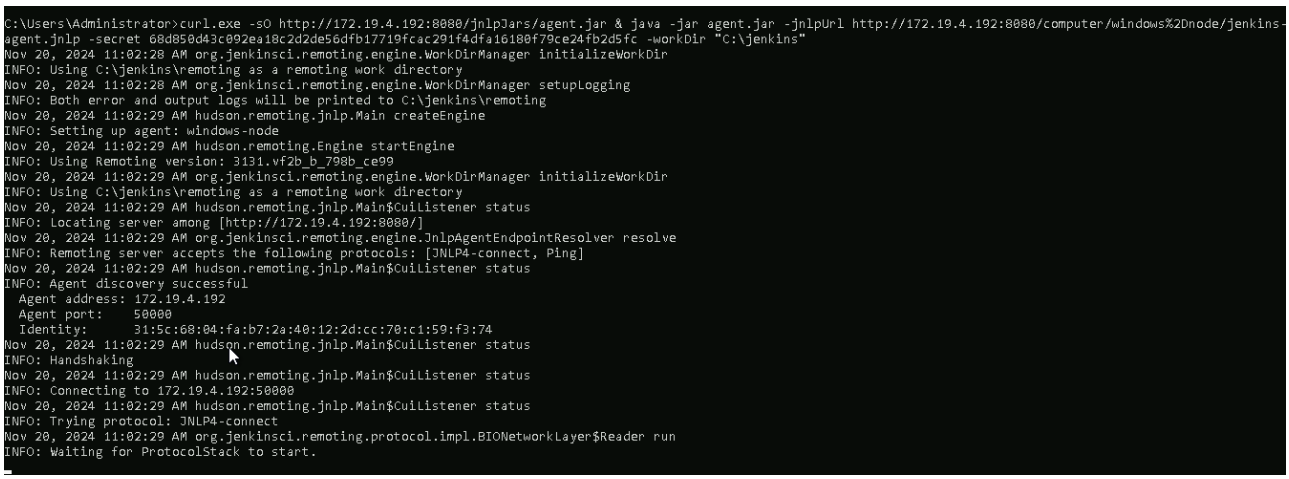
Step 3: Start the Jenkins Agent on the Windows Node

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**.



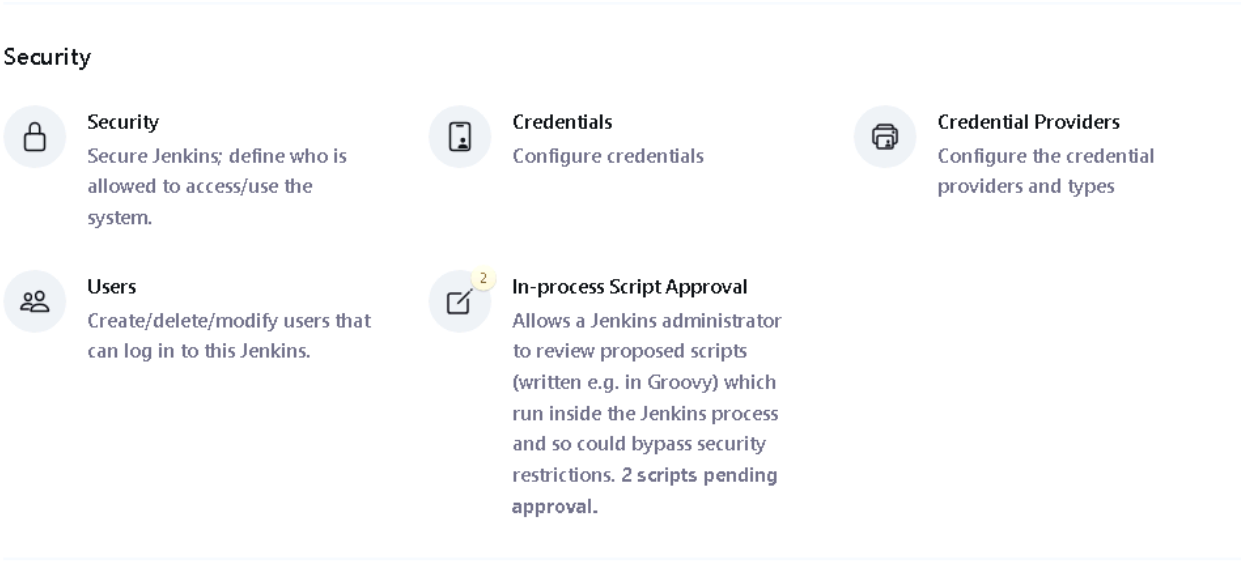


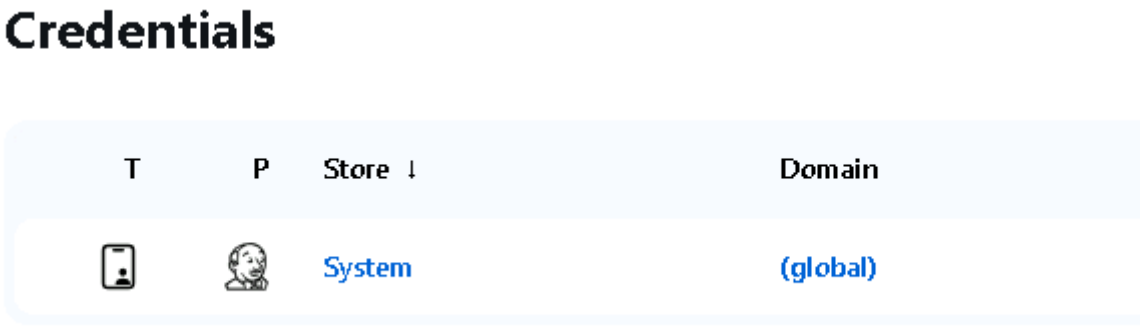
3. **Verify Connection**:
    - After running the command, the Jenkins agent should connect to the Jenkins master.
    - The node will show as **online** on the **Manage Nodes** page in Jenkins.

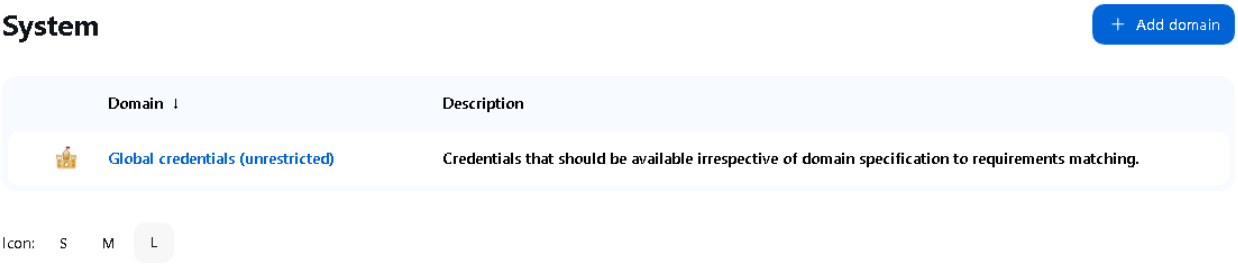
Step 4: Create and Configure Credentials in Jenkins

4.1 Add Chef Server Credentials

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





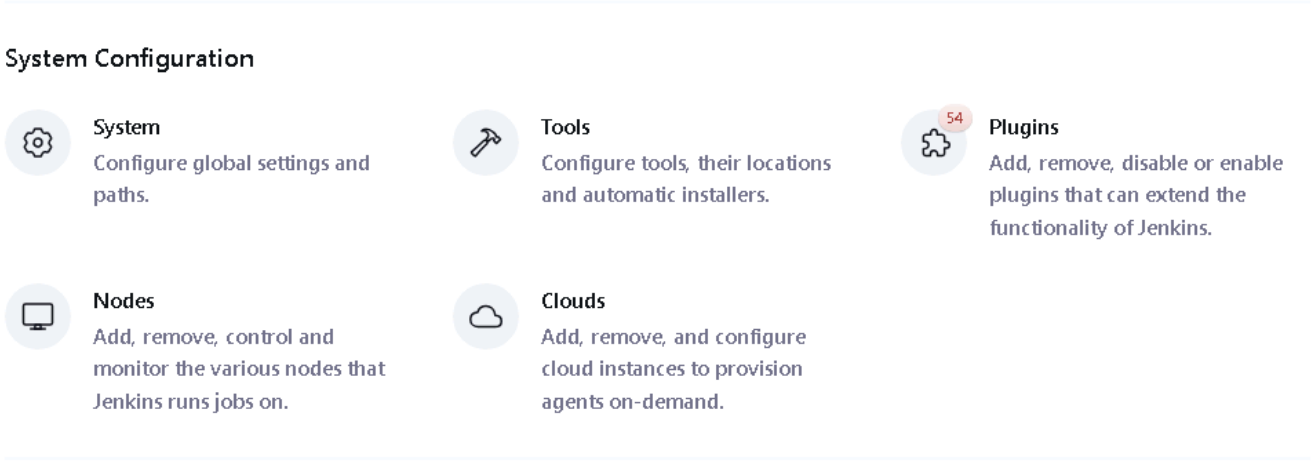


- Add a new credential:
  - **ID**: `chef-server-id`
  - **Username**: Chef Server username
  - **Password**: Chef Server password

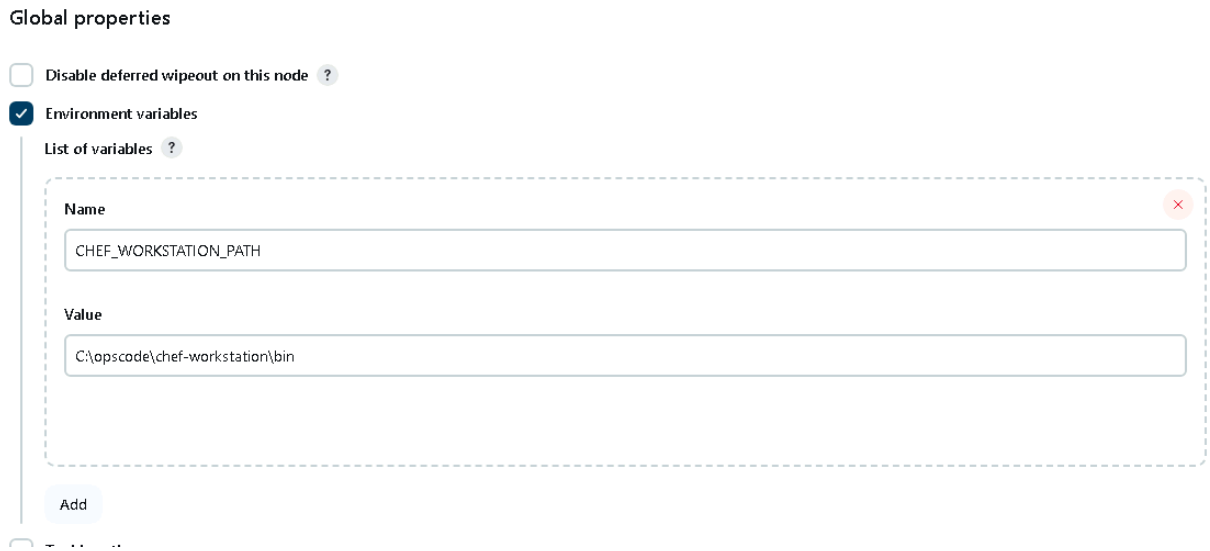
Step 5: Add CHEF_WORKSTATION_PATH as a Global Environment Variable

5.1 Configure Global Properties

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



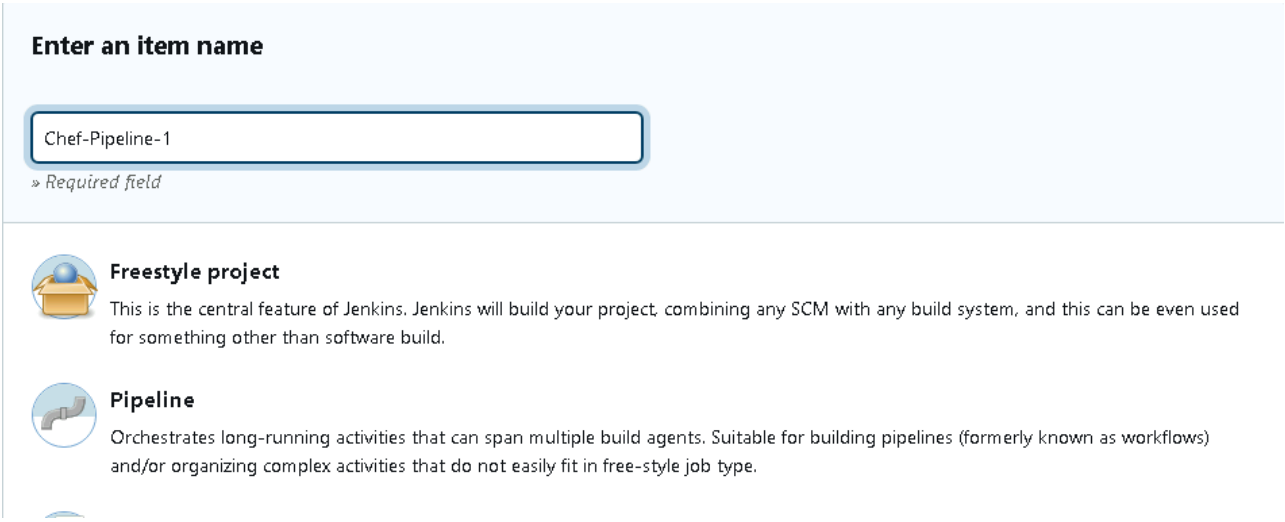
- Enable **Global Properties > Environment Variables**.
- Add the following variable:
  - **Name**: `CHEF_WORKSTATION_PATH`
  - **Value**: `C:/opscode/chef-workstation/bin`



Step 6: Run the Pipeline

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

- Create a Pipeline



- 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/