SLAVE DSIGN

Date

Author

Reviewr

Approver

~~Introduction to Jenkins master and slave~~

~~Design of Master and Slave~~

Introduction to docker

system configuration

Explanation of the setup

Design of Slave $ Docker setup

Slave Deep dive of File system and mount

conclusion

**1. Introduction to Jenkins master and slave**

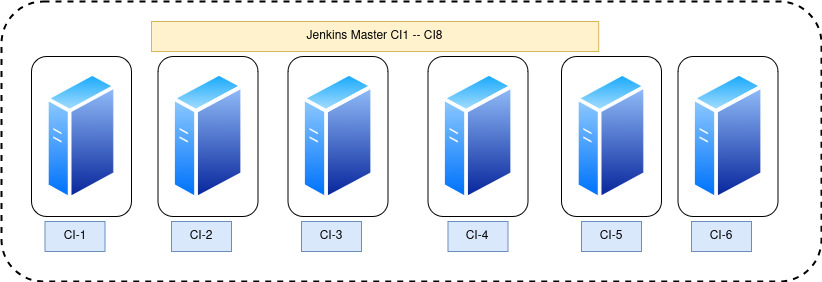
In Jenkins' Master-Slave architecture, the Master acts as the central control unit, managing configurations, scheduling jobs, and assigning tasks to slaves. The Slave (or agent) is a worker node that executes the actual tasks, such as building, testing, or deploying applications.

The master coordinates jobs, while slaves run them in parallel, enabling scalability and efficient workload distribution across different environments.

**A. Types of Slaves**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operating system** | **Version** | **Environment** | **No of Servers** |
| RHEL | 8.x | UAT | 18 |
| MS Windows | 2016 | UAT | 5 |

**Screenshot 1.0 Jenkins Master**

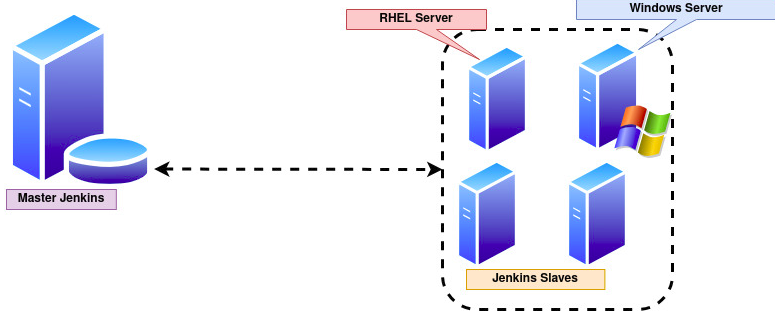


This diagram highlights the Jenkins Master setup, which serves as the central control unit in a CI/CD environment. The Master is responsible for managing job configurations, scheduling builds, monitoring pipelines, and coordinating all Jenkins activities. It handles tasks such as orchestrating builds and managing the user interface for administrators and developers.

In this design, the Master acts as the core of the system, ensuring jobs are properly triggered, tracked, and reported. It consolidates build results and provides a single point of control for the CI/CD pipeline, maintaining a streamlined and organized workflow.

***Note: This design has no Clustering which means single point of failure***

**Screenshot 1.1 Master to Slave connection**

****

***Note: The connection from Slave to Master is established using JNPL method***

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | **Responsibilities** |
| Master Jenkins | Central control unit of the Jenkins CI/CD pipeline. | - Configures and manages Jenkins jobs.  - Schedules builds and delegates tasks to slaves.  - Monitors job execution and consolidates results.  - Hosts the Jenkins web interface for management. |
| Jenkins Slaves | Worker nodes connected to the Master, responsible for executing delegated tasks. | - Executes builds, tests, and deployments as assigned by the Master.  - Reports job status and results back to the Master. |
| RHEL Server | Slave node configured to handle tasks requiring a Linux environment. | - Executes Linux-specific tasks such as shell scripting, Linux builds, or deployments. |
| Windows Server | Slave node configured to handle tasks requiring a Windows environment. | - Executes Windows-specific tasks such as .NET builds or Windows deployments. |
| Master-Slave Communication | Communication occurs through secure connections, with the Master assigning jobs and receiving results from slaves. | - Assigns jobs to slaves based on resource availability and job requirements.  - Receives and consolidates job statuses and outputs. |

**2. System Hardware Configuration**

|  |  |  |
| --- | --- | --- |
| O/S | CPU{core} | Memory |
| RHEL 8.x | 16 | 128(GB) |

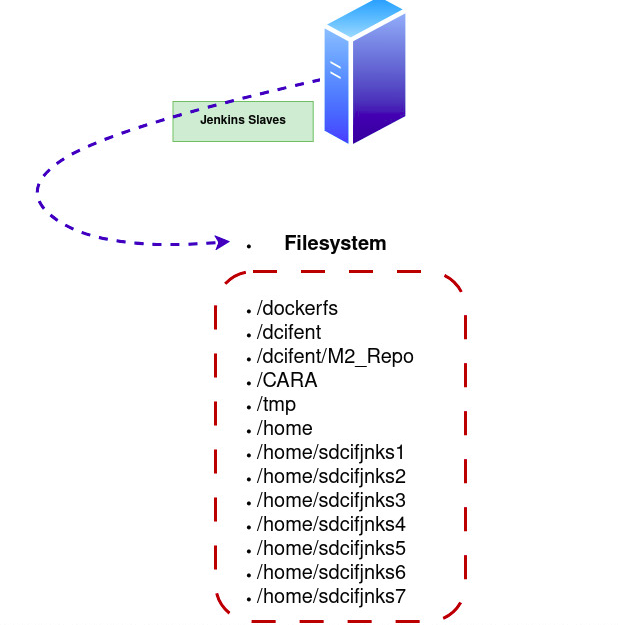
**2.1 System Disk / Mount Configuration**

|  |  |  |
| --- | --- | --- |
| Mount Name | Size (GB) | Type |
| / | 63 | Ext4 |
| /dockerfs | 295 | Ext4 |
| /dcifent | 609 | Ext4 |
| /dcifent/M2\_Repo | 285 | Ext4 |
| /CARA | 9.8 | Ext4 |
| /tmp | 20 | Ext4 |
| /home | 15 | Ext4 |
| /home/sdcifjnks1 | 10 | Ext4 |
| /home/sdcifjnks2 | 10 | Ext4 |
| /home/sdcifjnks3 | 10 | Ext4 |
| /home/sdcifjnks4 | 10 | Ext4 |
| /home/sdcifjnks5 | 10 | Ext4 |
| /home/sdcifjnks6 | 10 | Ext4 |
| /home/sdcifjnks7 | 10 | Ext4 |

**2.1 User & Group Information**

|  |  |  |
| --- | --- | --- |
| User Account | Group Information | Privilege {root / non root } |
| sdcifjnks1 | Docker | Non-Admin |
| sdcifjnks2 |
| sdcifjnks3 |
| sdcifjnks4 |
| sdcifjnks5 |
| sdcifjnks6 |
| sdcifjnks7 |
| sdcifjnks8 |
| tdcifadm | N/A | Non-Admin |

**Screenshot 1.2 Slave Design**

****

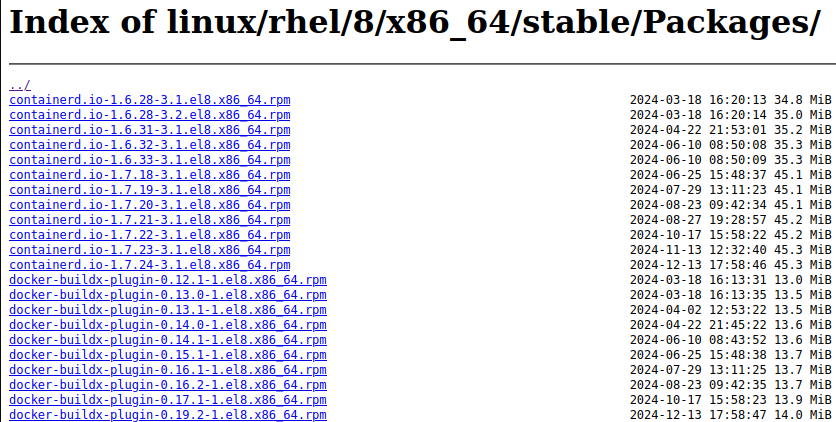
**3.Introduction to Docker**

Docker is a platform used for building containerized images that can be deployed on various container-supporting platforms, such as OpenShift, AWS, VMware VIC, Azure, and Kubernetes, with a primary focus on OpenShift environments. It plays a vital role in packaging applications and their dependencies into lightweight, portable containers, ensuring consistency across environments. Docker is essential for building, compiling, and deploying applications within images, enabling efficient workflows, scalability, and cross-platform compatibility.

***Note:- Non rooted Daemon is under POC which is the best practice for docker execution***

**3.1 Setting up Docker**

The installation is performed using manual method as the RHEL Repo has no official binary from REDHAT However we are using official rpm from docker **ht{t}ps[:]//download.docker.com/linux/rhel/8/x86\_64/stable/Packages/**



(a) The docker yum repo has been whitelisted in GAAS and has been mirrored in Jforg this will help us to setup the docker on the **linux** server using dnf command line

(b) The update / Upgrade can be easly managed as the repo is configured locally same as redhat.repo

**Command to Setup the Docker**

a. dnf list repo ( should display docker repo )

b. dnf install docker-ce docker-compose docker-cli containerd.io -y

c. systemctl stop docker docker.socket

St

Explanation of the setup

Design of Slave $ Docker setup

Slave Deep dive of File system and mount

***conclusion***