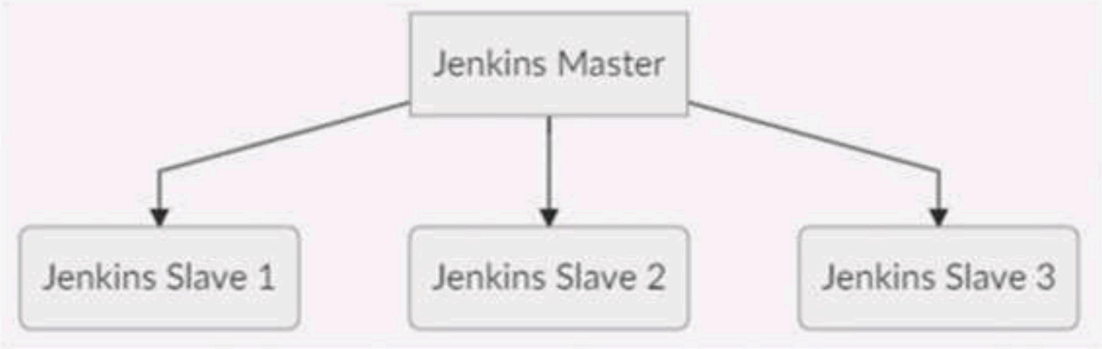
Aim: To understand Jenkins Master-Slave Architecture and scale your Jenkins standalone implementation by implementing slave nodes**.**

Objective: The objective of understanding Jenkins Master-Slave architecture is to comprehend how to scale Jenkins infrastructure by implementing slave nodes, thereby distributing the workload and enhancing the performance and resilience of the Jenkins CI/CD system

Theory:

A Jenkins master comes with the basic installation of Jenkins, and in this configuration, the master handles all the tasks for your build system. You are working on multiple projects, you may run multiple jobs on each project. Some projects need to run on some nodes, and in this process, we need to configure slaves. Jenkins slaves connect to the Jenkins master using the Java Network Launch Protocol.
Jenkins Master and Slave Architecture



The Jenkins master acts to schedule the jobs, assign slaves, and send builds slaves to execute the jobs. It will also monitor the slave state (offline or online) and get back the build result responses from slaves and the display build results on the console output. The workload of building jobs is delegated to multiple slaves.

Steps:

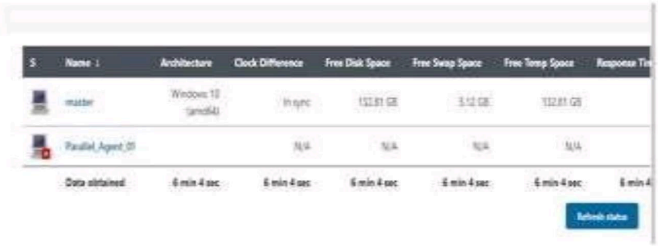
1. Click on Manage Jenkins in the left corner on the Jenkins dashboard.
2. Scroll down, Click on Manage Nodes and clouds.

3. Select New Node and enter the name of the node in the Node Name field.

Select **New Node** and enter the name of the node in the Node Name field.

4. Select Permanent Agent and click the OK button. Initially, you will get only one option, "Permanent Agent." Once you have one or more slaves you will get the "Copy Existing Node" option.

Select **Permanent Agent** and click the **OK** button. Initially, you will get only one option, "Permanent Agent." Once you have one or more slaves you will get the "Copy Existing Node" option.



In the above screen shot, Parallel_Agent_01 was Created and currently it is in offline mode.

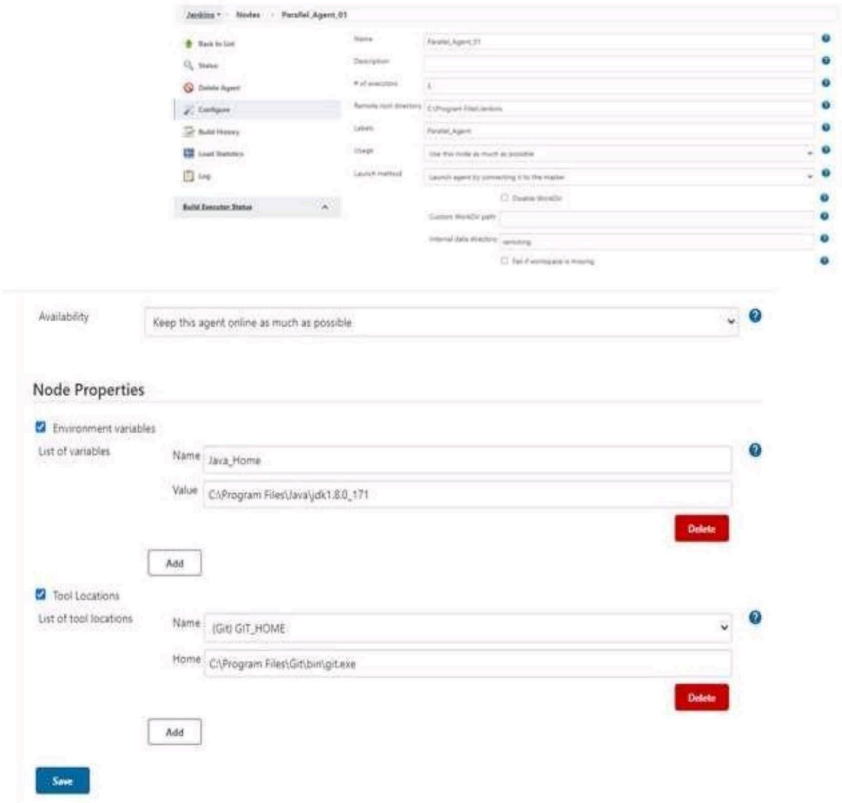
5. Click on configure, Provide the details.
   1. Name -Parallel_Agent_01.
   2. Number of executors- 5.
   3. Remote root directory-We have to provide a Jenkins path.
   4. Labels-Parallel_Agent.
   5. Launch method-Launch agent by connecting it to the master.

6. Node Properties Tab:

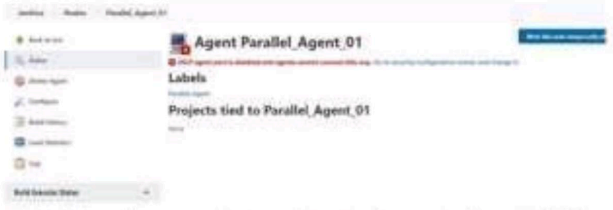
• Check Environment variables Provide the Java path

• Check Tool Locations

Provide the Git path and click on save button.

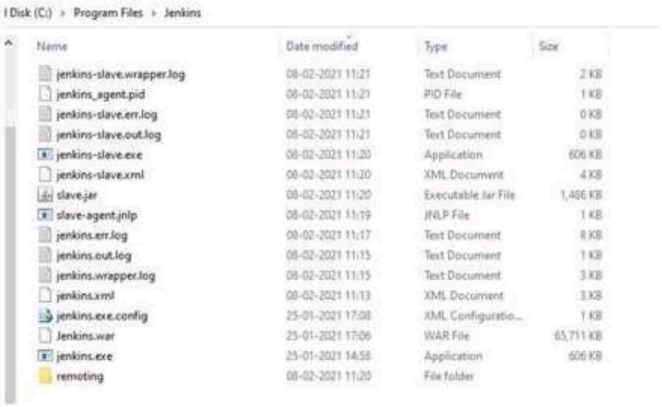


7. Click on Go to the security configuration screen and change it. It will redirect to Configure Global Security → Agents > click on Fixed radio button port: 49187 and click on save Button. Go back to Nodes settings.



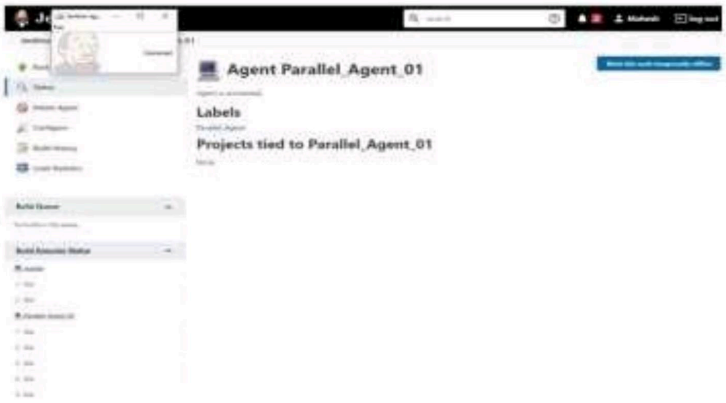
8. We can see the screen,
    1. Click on Launch button, it will download the launch agent in your system.

    2. Jenkins-slave.exe file should copy in the Jenkins folder which you installed in your system.
    3. Double Click on jenkins - slave.exe.

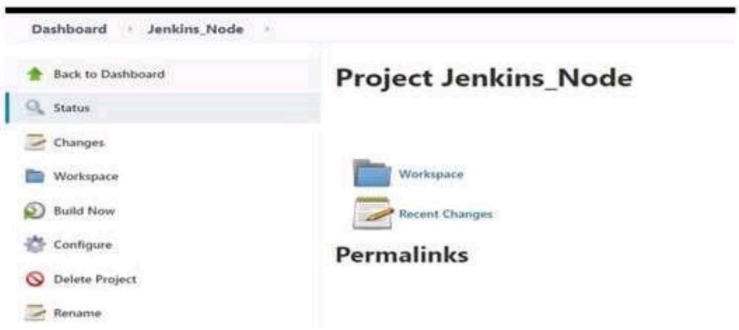


    4. Run the launch agent, click on the run button and it will show connected.

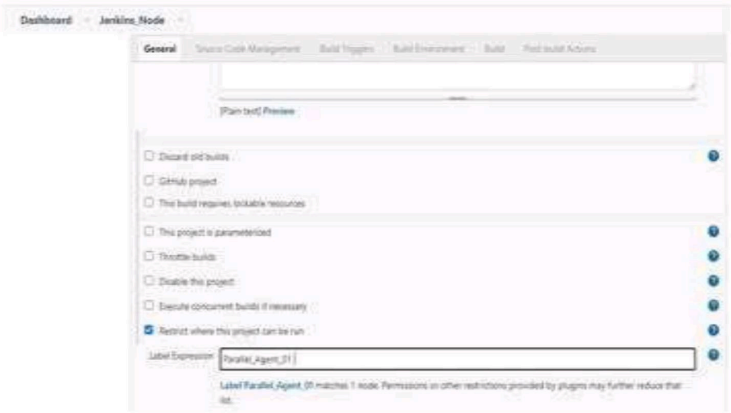
5. In the screenshot below, we can see the connected popup, click on the file menu, select the install as service and click yes button. Once it is done, refresh the page.



9. we can see the Build executors. One is master and other is Parallel_agent_01 In Master node, we can see the number of executors as 2.
   - In Parallel_agent_01, we can see the number of executors as 5.
   - Go to build job -> configure.



- In the General tab, check on Restrict where this project can be run.
- In Label Expression, we have to select the node name where we need to execute the build job.



- We can create more nodes as well.

Conclusion:

Q1. How does Jenkins communicate between master and slave?

Jenkins uses a Master-Slave architecture to manage distributed builds. The communication between the Master and Slave nodes is carried out through TCP/IP protocol. The Master node is responsible for scheduling build jobs, assigning slaves, and sending builds to slaves to execute the job. The Slave nodes, also known as Agents, listen to commands from the Master and execute build jobs dispatched by the Master. The communication protocol used for this Interaction is Jenkins Remoting, a binary format. By default, the communication between the Master and Slave nodes is not encrypted[1]. However, there are options to encrypt the communication, such as using JNLP (Java Network Launch Protocol) with SSL (Secure Sockets Layer) or TLS (Transport Layer Security).The communication mechanism between the Master and Slave nodes is described in the

Jenkins Remoting source code. The Master initiates the connection to the Slave, and after a successful connection, the Slave sends its key to the Master for authentication and encryption.

Q2. How many slaves can be connected to Jenkins master?

Jenkins allows for multiple slave nodes to be connected to a master node, enabling distributed builds and testing. The number of slaves that can be connected to a Jenkins master is not limited to a specific number. Instead, the scalability of Jenkins allows for a flexible setup where you can connect as many slave nodes as needed to handle the workload efficiently. This architecture is particularly useful in scenarios where different builds require different environments or when there is a high volume of builds to be executed. By configuring and connecting multiple slave nodes to a Jenkins master, organizations can optimize their build processes and improve overall efficiency.