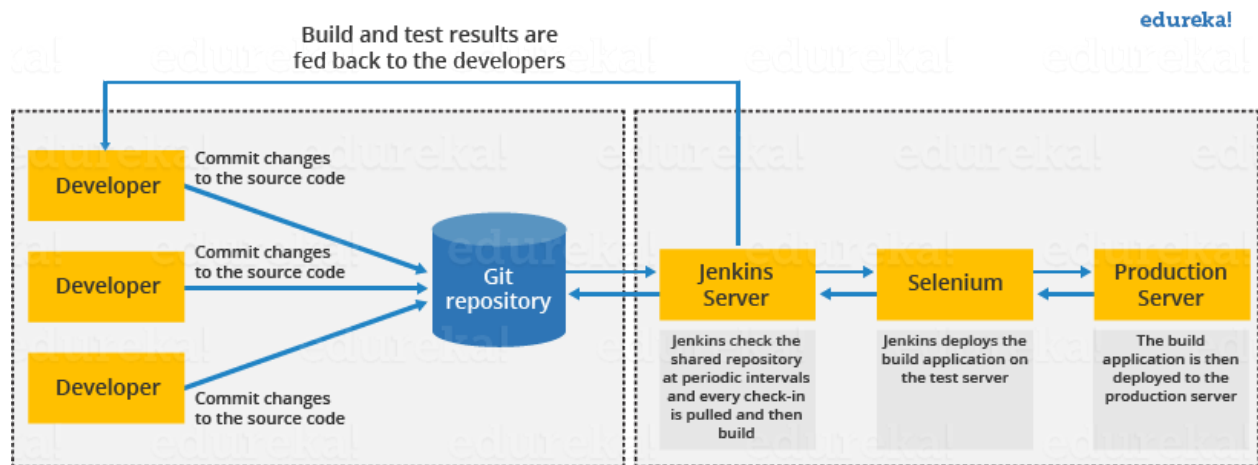


Experiment 07 - Jenkins Architecture

Roll No.	24
Name	Sreekesh Iyer
Class	D15-A
Subject	DevOps Lab
LO Mapped	<p>LO1: To understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet your business requirements</p> <p>LO3: To understand the importance of Jenkins to Build and deploy Software Applications on server environment</p>

Aim:

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

Architecture:**Jenkins Architecture**

This single Jenkins server was not enough to meet certain requirements like:

- Sometimes you might need several different environments to test your builds. This cannot be done by a single Jenkins server.
- If larger and heavier projects get built on a regular basis then a single Jenkins server cannot simply handle the entire load.

To address the above-stated needs, Jenkins distributed architecture came into the picture.

Jenkins Distributed or Master-Slave Architecture

Jenkins uses a Master-Slave architecture to manage distributed builds. In this architecture, Master and Slave communicate through TCP/IP protocol.

Jenkins Master:

Your main Jenkins server is the Master. The Master's job is to handle:

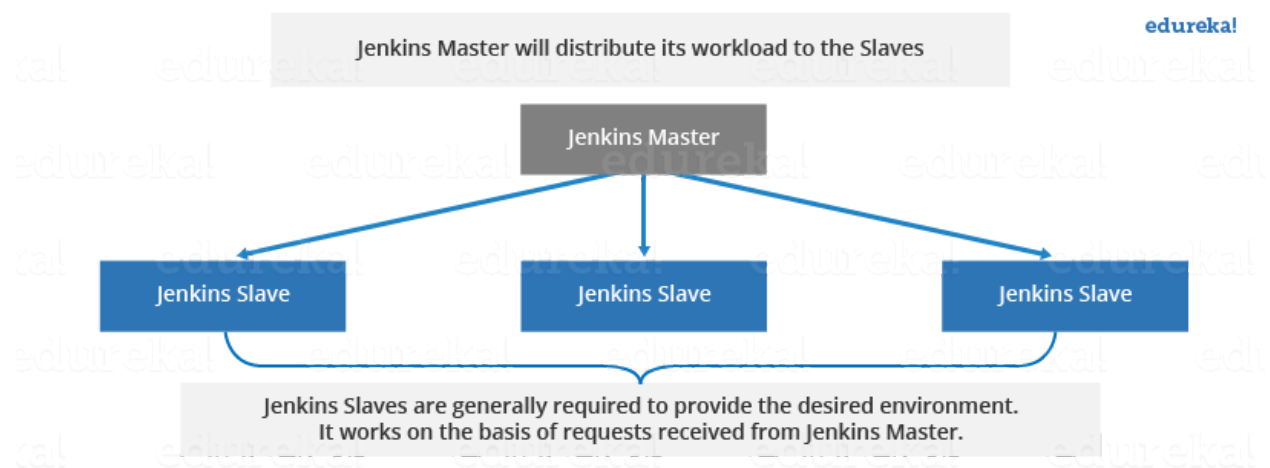
- Scheduling build jobs.
- Dispatching builds to the slaves for the actual execution.
- Monitor the slaves (possibly taking them online and offline as required).
- Recording and presenting the build results.
- A Master instance of Jenkins can also execute build jobs directly.

Jenkins Slave:

A Slave is a Java executable that runs on a remote machine. The characteristics of any Jenkins Slaves are:

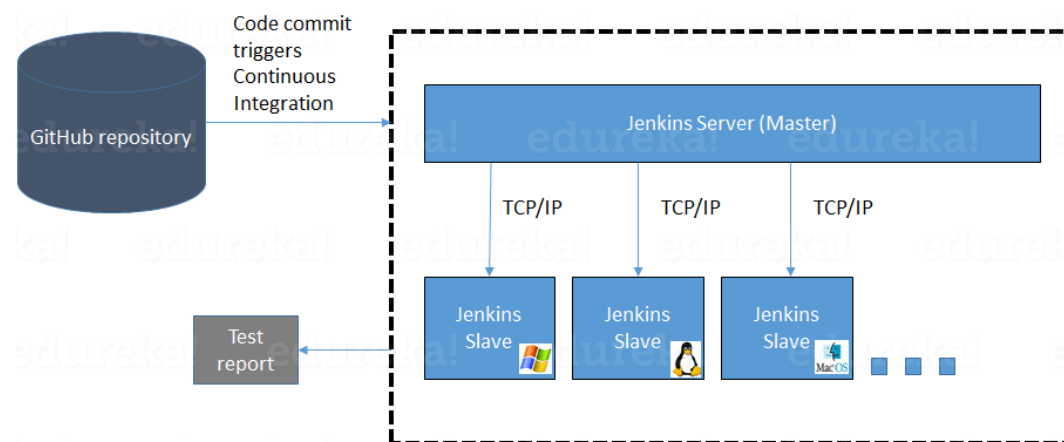
- It hears requests from the Jenkins Master instance.
- Slaves can run on a variety of operating systems.
- The job of a Slave is to do as they are told to, which involves executing build jobs dispatched by the Master.
- You can configure a project to always run on a particular Slave machine or a particular type of Slave machine, or simply let Jenkins pick the next available Slave.

The diagram below is self-explanatory. It consists of a Jenkins Master which manages three Jenkins Slaves.



How does Jenkins Master and Slave Architecture work?

The diagram below represents the use of Jenkins for testing in different environments like Ubuntu, MAC, Windows, etc.



The above image represents the following functions:

- Jenkins checks the Git repository at periodic intervals for any changes made in the source code.
- Each build requires a different testing environment which is not possible for a single Jenkins server. In order to perform testing in different environments, Jenkins uses various Slaves as shown in the diagram.
- Jenkins Master requests these Slaves to perform testing and to generate test reports.

Scalability

Scalability is an essential component of enterprise software. Prioritizing it from the start leads to lower maintenance costs, better user experience, and higher agility. Software design is a balancing act where developers work to create the best product within a client's time and budget constraints. There's no avoiding the necessity of compromise. Tradeoffs must be made in order to meet a project's requirements, whether those are technical or financial. Too often, though, companies prioritize cost over scalability or even dismiss its importance entirely. This is unfortunately common in big data initiatives, where scalability issues can sink a promising project.

Benefits of Scalable Software

Scalability has both long- and short-term benefits. At the outset, it lets a company purchase only what they immediately need, not every feature that might be useful down the road.

For example, a company launching a data intelligence pilot program could choose a massive enterprise analytics bundle, or they could start with a solution that just handles the functions they need at first. A popular choice is a dashboard that pulls in results from their primary data sources and existing enterprise software. When they grow large enough to use more analytics programs, those data streams can be added to the dashboard instead of forcing the company to juggle multiple visualization programs or build an entirely new system. Building this way prepares for future growth while creating a leaner product that suits current needs without extra complexity.

It requires a lower up-front financial outlay, too, which is a major consideration for executives worried about the size of big data investments. Scalability also leaves room for changing priorities. That off-the-shelf analytics bundle could lose relevance as a company shifts to meet the demands of an evolving marketplace. Choosing scalable solutions protects the initial technology investment. Businesses can continue using the same software for longer because it was designed to grow along with them. When it comes time to change, building onto solid, scalable software is considerably less expensive than trying to adapt less agile programs.

There's also a shorter "ramp up" time to bring new features online than to implement entirely new software. As a side benefit, staff won't need much training or persuasion to adopt that

upgraded system. They're already familiar with the interface, so working with the additional features is viewed as a bonus rather than a chore.

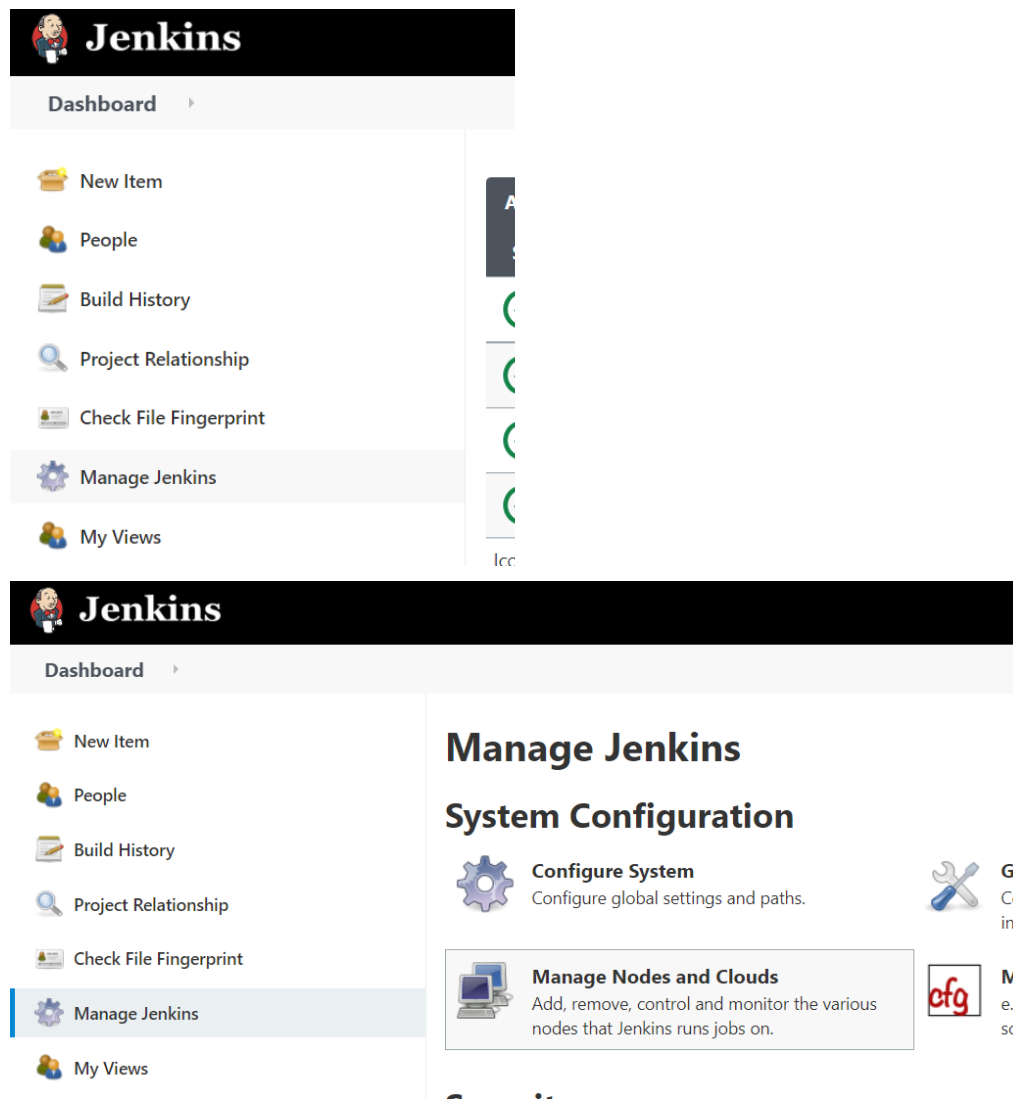
Jenkins Scalability

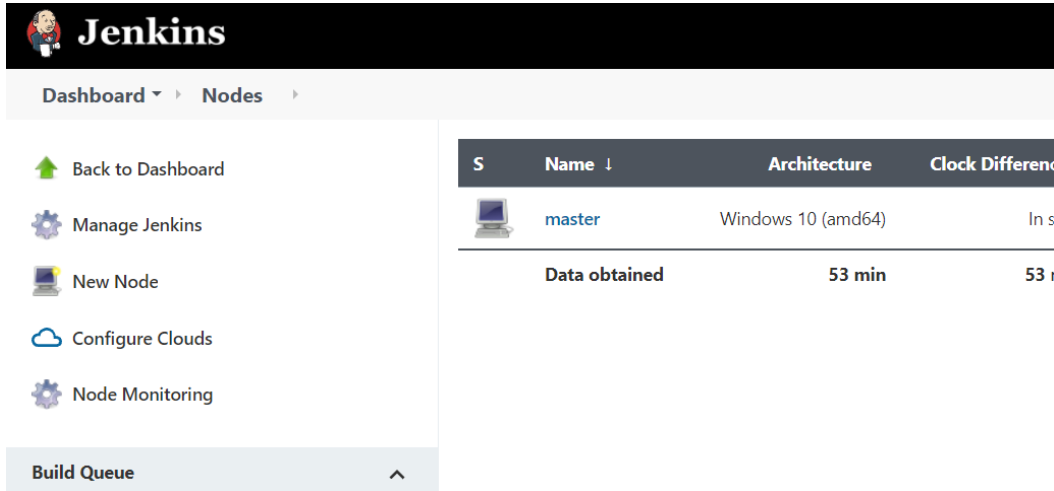
One of the strongest features that Jenkins has is scaling—pushing beyond its normal limits of productivity. In this article, we're taking a look at the Jenkins master/slave model with one central Jenkins instance, referred to as master, and a couple of slave executables called slaves, with the master responsible for scheduling the jobs across the slaves.

Scaling:

Scaling in Jenkins corresponds to creating master and slave nodes.

Step 1: Go to the Manage Jenkins section and scroll down to the section of Manage Nodes.

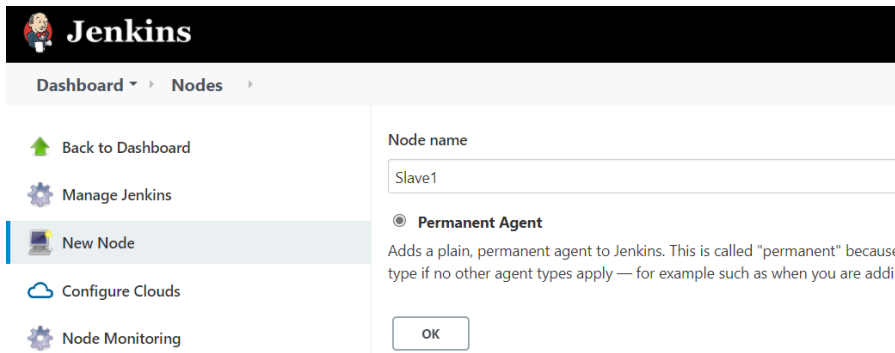


Step 2: Click on New Node

The screenshot shows the Jenkins 'Nodes' page. On the left is a sidebar with navigation links: 'Back to Dashboard', 'Manage Jenkins', 'New Node', 'Configure Clouds', and 'Node Monitoring'. The 'New Node' link is highlighted. The main area displays a table of nodes. The table has columns: 'S', 'Name', 'Architecture', and 'Clock Difference'. There is one node listed: 'master' with architecture 'Windows 10 (amd64)'. Below the table, there is a section for 'Data obtained' showing '53 min' and '53 r'.

S	Name	Architecture	Clock Difference
	master	Windows 10 (amd64)	In s

Data obtained: 53 min, 53 r

Step 3: Give a name for the node, choose the Permanent Agent option and click on Ok.

The screenshot shows the 'New Node' form in Jenkins. The 'Node name' field is filled with 'Slave1'. The 'Permanent Agent' option is selected. Below the form, there is a description: 'Adds a plain, permanent agent to Jenkins. This is called "permanent" because type if no other agent types apply — for example such as when you are addir'. An 'OK' button is at the bottom.

Node name: Slave1







☒ Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because type if no other agent types apply — for example such as when you are addir

OK

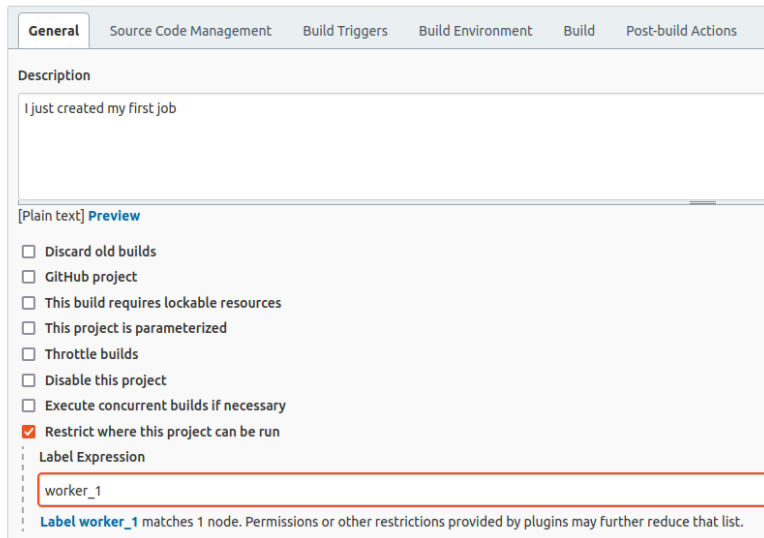
Step 4: Enter the details of the node slave machine.

1. Here no. of executors in nothing but no. of jobs that this slave can run parallely. Here we have kept it to 2.
2. The Labels for which the name is entered as "Slave1" is what can be used to configure jobs to use this slave machine.
3. Select Usage to Use this node as much as possible.
4. For launch method we select the option of "Launch agent by connecting it to the master".
5. Here in the Agents section click on Random and Save it. Now you will find the required option.
6. Enter Custom WorkDir path as the workspace of your slave node.
7. In Availability select "Keep this agent online as much as possible".
8. Click on Save.

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	master	Linux (amd64)	In sync	4.91 GB	252.46 MB	4.91 GB	0ms 
	worker-1	Linux (amd64)	In sync	4.91 GB	252.46 MB	4.91 GB	200ms 
	worker-2	Linux (amd64)	In sync	4.91 GB	252.46 MB	4.91 GB	181ms 
Data obtained		0.28 sec	0.37 sec	0.19 sec	0.17 sec	0.19 sec	0.24 sec
<div>Refresh status</div>							

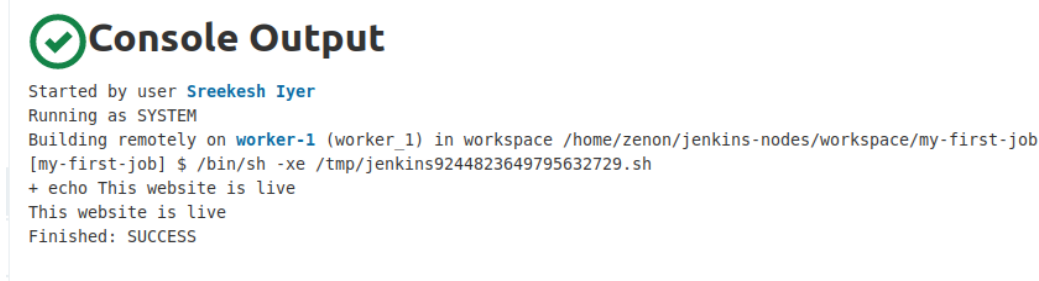
Step 6: Now since your slave agent is up and running, let's execute a job on it.

1. For that I already have an existing job and I will run this job on this slave. Open this job and click on configure.
2. Now here in the General section, click on “Restrict where this project can be run”.
3. Here in Label Expression, enter the name of the slave and save it.
4. Click Save



The screenshot shows the Jenkins job configuration interface. The 'General' tab is selected. Under the 'Description' field, it says 'I just created my first job'. Below this, there are several checkboxes for build options. The checkbox 'Restrict where this project can be run' is checked. Below this checkbox, the 'Label Expression' field is visible, containing the text 'worker_1'. A message below the field states: 'Label worker_1 matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.'

Step 7: Now click on Build now and see the output of this job. If everything is correct you will see the output as Success.



The screenshot shows the Jenkins Console Output. It starts with a green checkmark icon and the text 'Console Output'. Below this, it shows the build details: 'Started by user Sreekesh Iyer', 'Running as SYSTEM', 'Building remotely on worker-1 (worker_1) in workspace /home/zenon/jenkins-nodes/workspace/my-first-job', '[my-first-job] \$ /bin/sh -xe /tmp/jenkins9244823649795632729.sh', '+ echo This website is live', 'This website is live', and 'Finished: SUCCESS'.

Conclusion

Thus, we have understood what Jenkins Master and Slave Architecture is along with the importance of scalability while developing software. We also learnt the steps to show scalability in our Jenkins setup by creating new slave nodes.