CI/CD Pipeline with Jenkins

Distributed System in Jenkins

Learning Objectives

By the end of this lesson, you will be able to:

- Explain distributed architecture
- Set up master/slave Jenkins instances
- Generate and assign tasks to slave machines
- Explain SSH and Java Web Start processes to start the slave machine

FULL STACK

Introduction to Distributed Architecture

Introduction

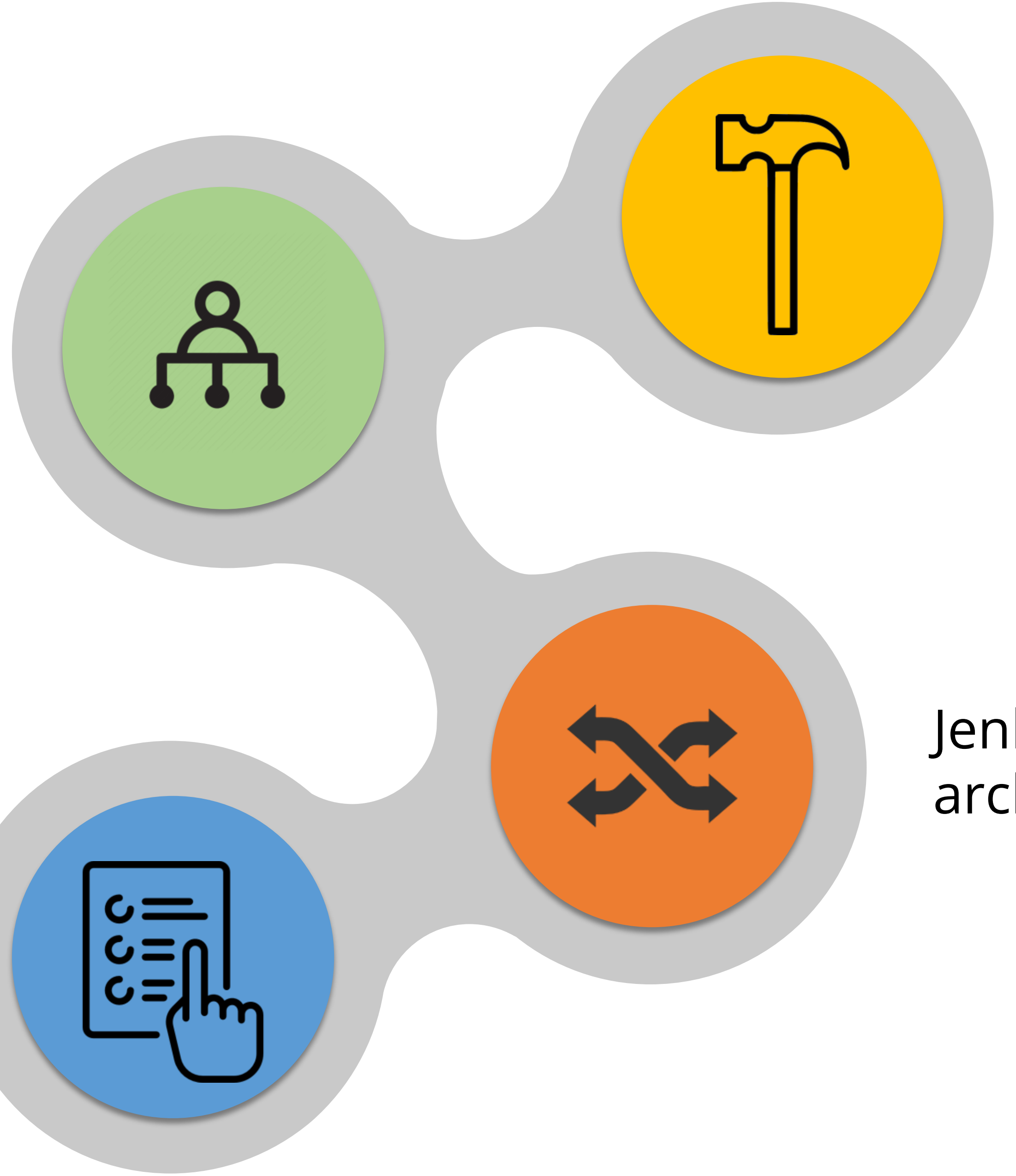
One of the more important features of Jenkins is the capability to execute and dispatch build jobs over a large number of systems.

- Build servers are created to either share the load across multiple machines, or to run build jobs in different environments.
- The need of build servers can vary during the project life cycle.
- For example, if you are working with product release cycles, run a higher number of builds jobs at the end of the cycle for comprehensive tests.

Jenkins Distributed Build Architecture

The role of Jenkins master's job is to schedule build jobs, assign builds to the slaves, record and generate build results.

Major characteristics of distributed architecture:



Jenkins slaves' tasks are decided by the Jenkins master.

Jenkins master instance can directly execute build jobs.

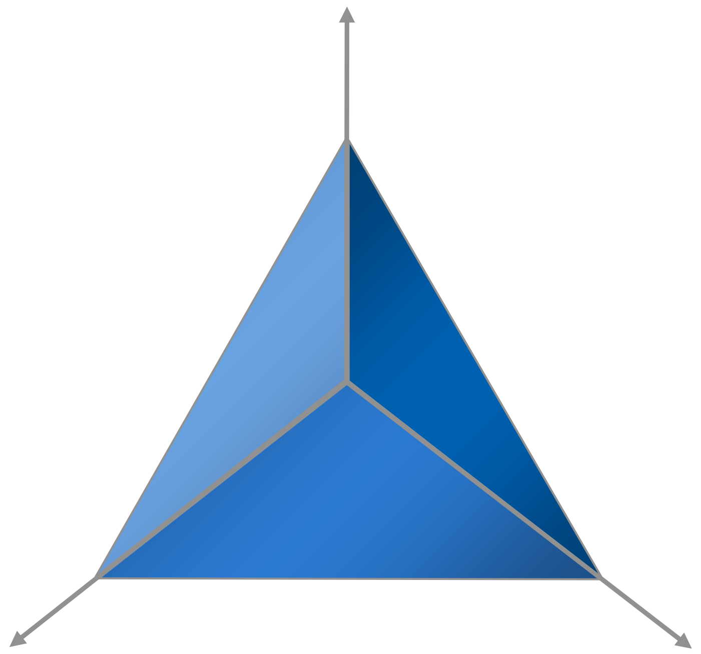
Jenkins master-slave architecture is highly flexible.

A project can run on a particular slave machine, on a particular type of machine, or let Jenkins pick the available slave.

Jenkins Slave Machine

A slave is a small Java file that runs on a remote machine and is assigned tasks by Jenkins master machine. Major characteristics of a slave machine include:

Slave machines can run on a variety of operating systems.



Slave machines communicate with the master instance through a TCP/IP connection.

The working mechanism of the slave machines depends on the operating system and network architecture.

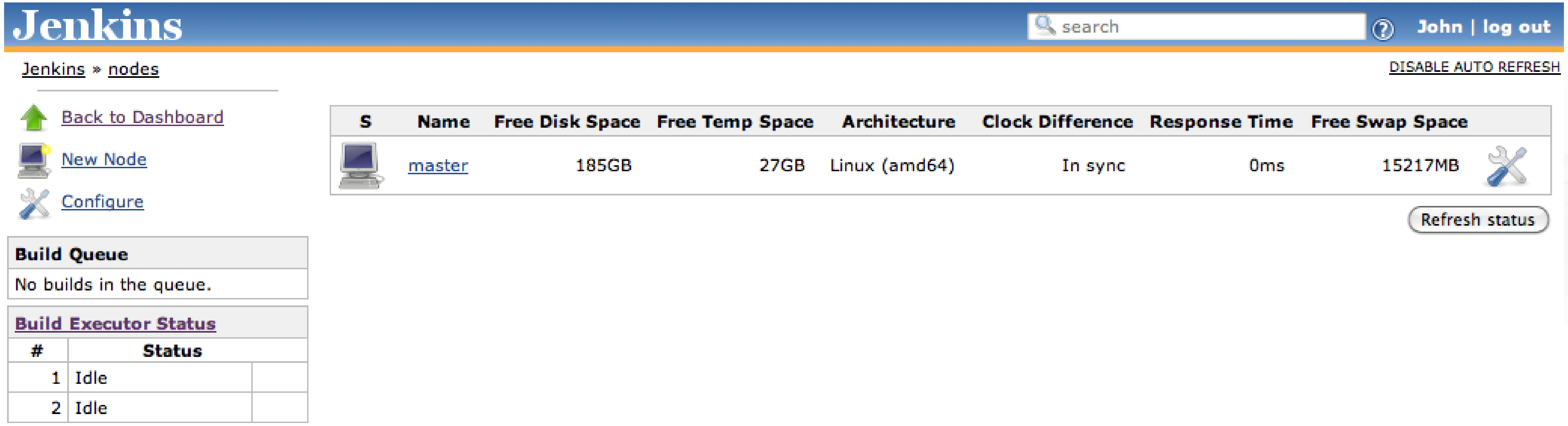
Master/Slave Strategies in Jenkins

A build job, running on a slave machine, is accessible by the end-user. Moreover, the build results always get generated and published on the master server.

- Creating a new Jenkins slave node is a straight-forward process and includes the following steps:

1. Navigate to **Manage Jenkins** screen and click on **Manage Nodes**.

1. Click on **New Node.**



S	Name	Free Disk Space	Free Temp Space	Architecture	Clock Difference	Response Time	Free Swap Space
	master	185GB	27GB	Linux (amd64)	In sync	0ms	15217MB

Build Queue

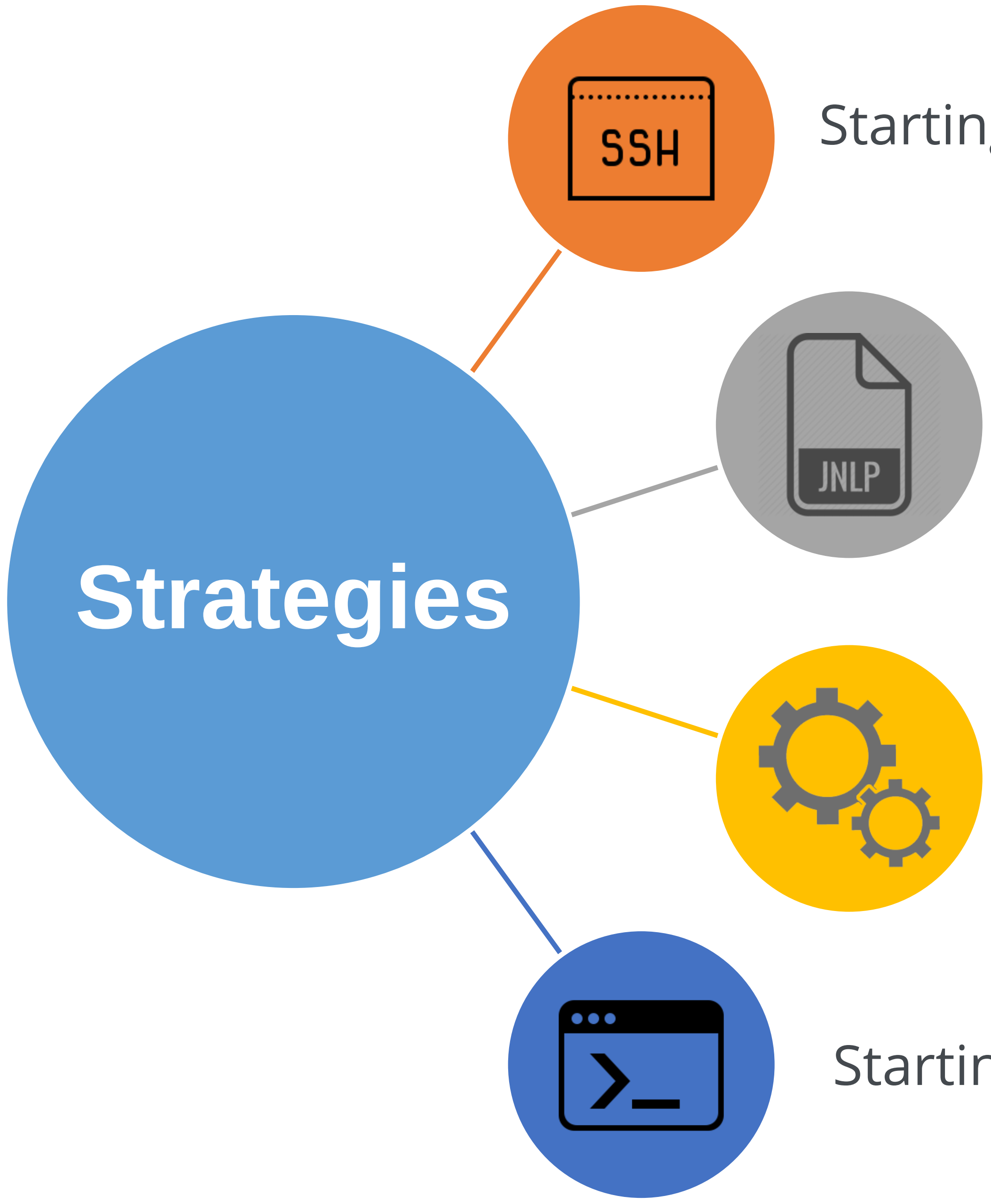
No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

Master/Slave Strategies in Jenkins

The four strategies of Jenkins master/slave architecture are:

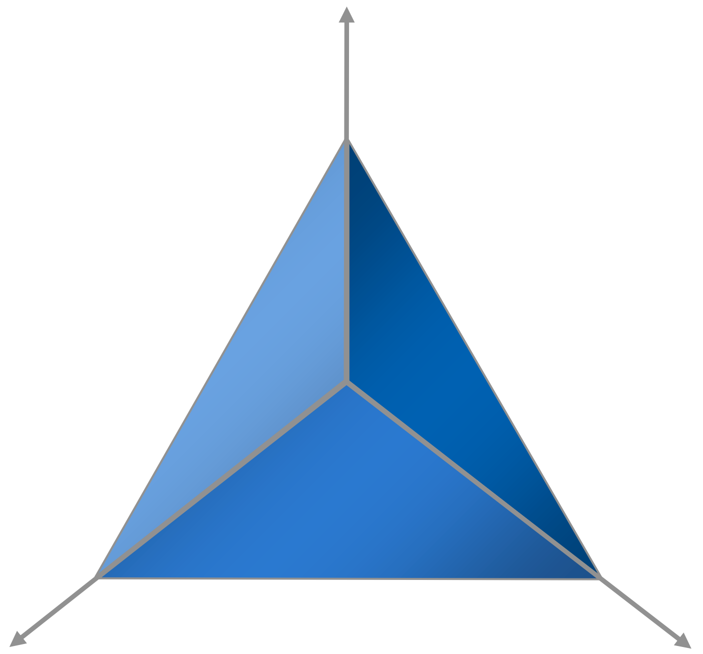


- Starting the slave machine with SSH connection
- Starting the slave machine manually with Java Web Start
- Installing the slave machine as a Windows service
- Starting the slave machine using CLI on the slave machine

Why Label Node?

Labels are tags to a slave node that allows node categorization in Jenkins.
Importance of labelling include:

Labels give information about the tools supported by a node.



Labels also provide information about the operating system of the node.

Labels can be kept for slaves in different locations due to multiple factors like different datacenters or cloud stack.

Create Multiple Slave Node

Duration: 30 mins.

Problem Statement: You are given a project to create and configure a slave node in Jenkins.

Steps to perform:

1. Spin up a VM to be used as the slave machine.
2. Install Jenkins on the VM.
3. Login to Master Node.
4. Navigate to Manage Jenkins.
5. Click on Manage Node and Clouds.
6. Click on New Node.
7. Provide name, number of executors, root directory path, label, usage, launch method, and WebDir path.
8. Once the configuration is complete, you can see the slave machine on the dashboard.

ASSISTED PRACTICE

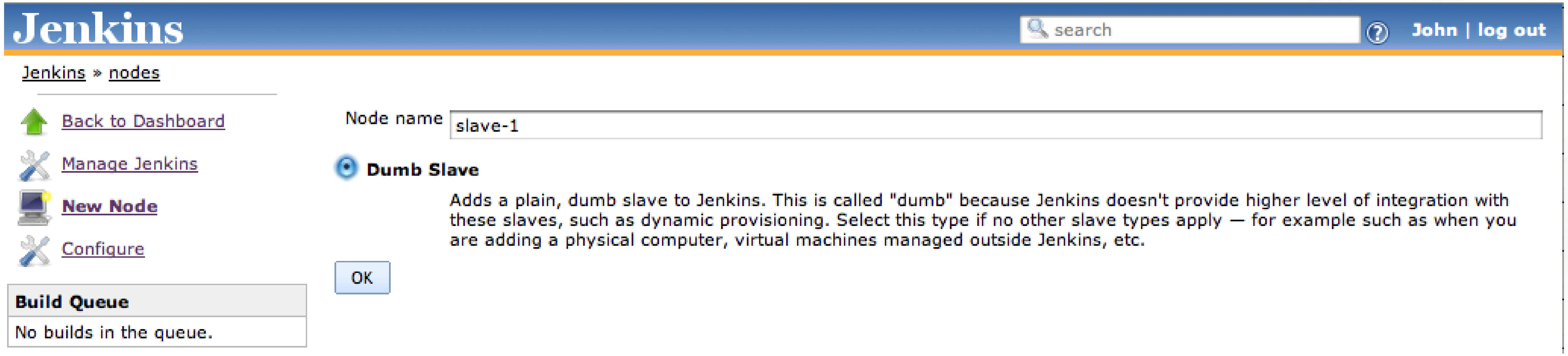
Deep Diving Jenkins Master/Slave Architecture

Starting Slave Machine with SSH

This strategy is useful for Unix machines as Jenkins has its own build-in SSH client.
Follow the below steps to create Unix-based slave:

1. Click on the **New Node** button.

1. Enter the name of your slave, and its type.



Build Queue
No builds in the queue.

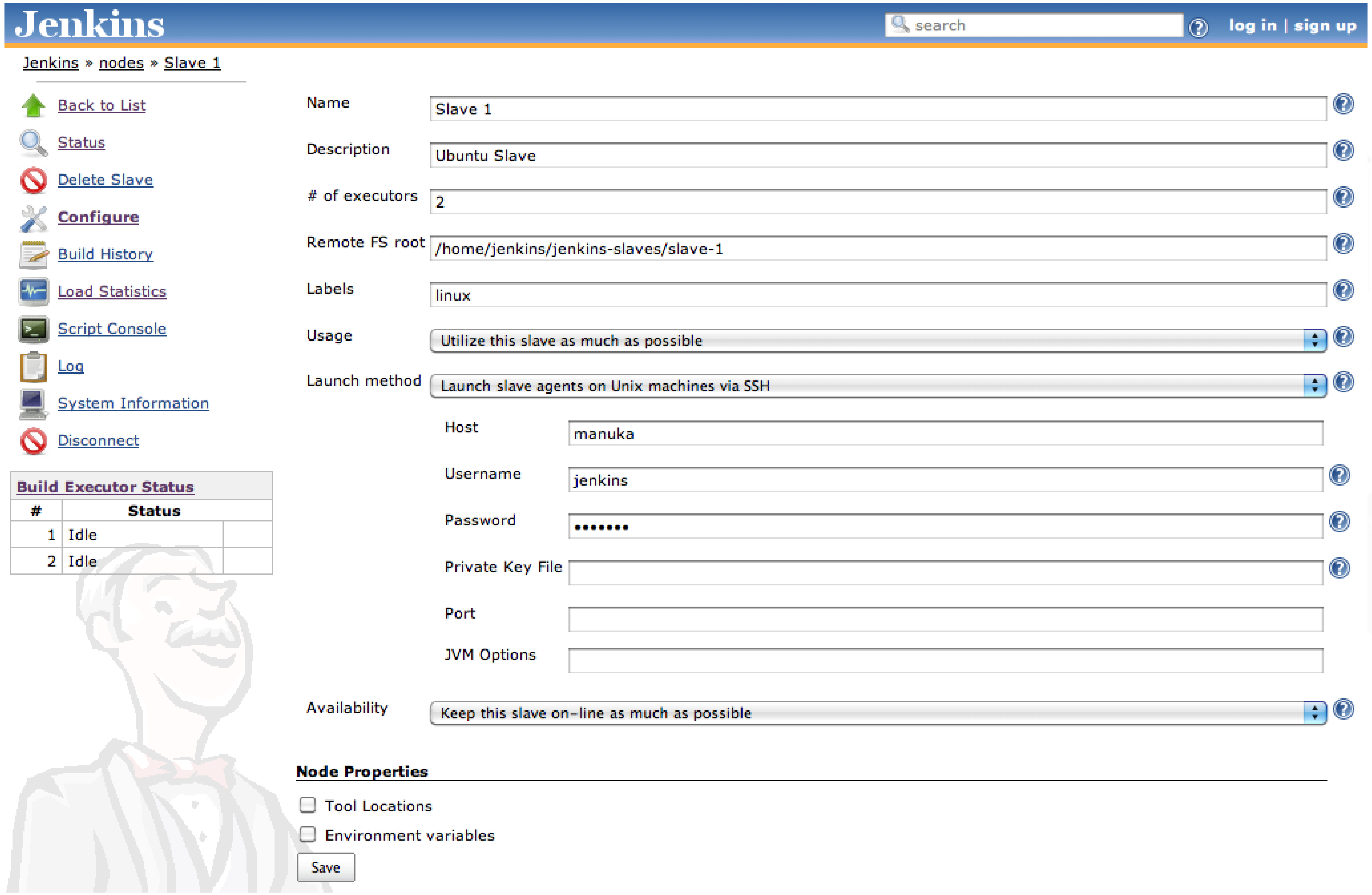
Node name: slave-1

[✓ Dumb Slave]

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

Configuring Slave Machine with SSH

The screenshot given below has the configuration fields of a Jenkins slave node.



Build Executor Status

#	Status
1	Idle
2	Idle

Name	Slave 1
Description	Ubuntu Slave
# of executors	2
Remote FS root	/home/jenkins/jenkins-slaves/slave-1
Labels	linux
Usage	Utilize this slave as much as possible
Launch method	Launch slave agents on Unix machines via SSH
Host	manuka
Username	jenkins
Password	•••••••
Private Key File	
Port	
JVM Options	
Availability	Keep this slave on-line as much as possible

Node Properties

- [ ] Tool Locations
- [ ] Environment variables

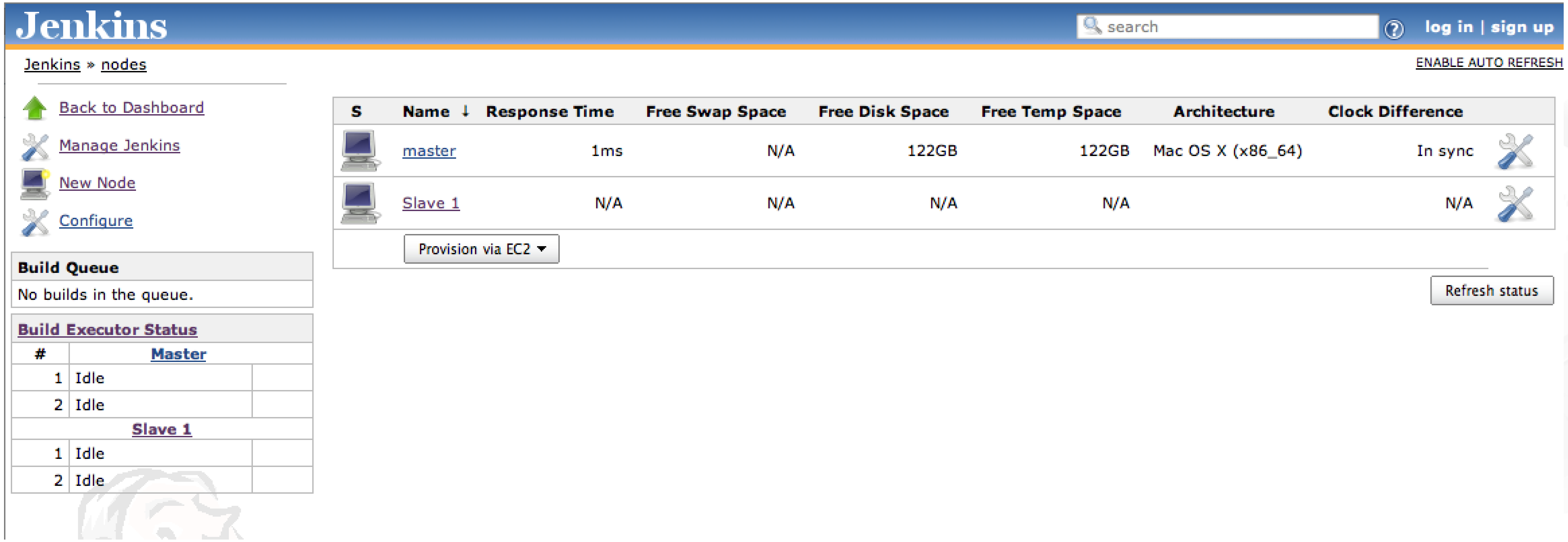
Save

Configuring Slave Machine with SSH

- The number of executors field lets you define the number of build jobs the particular node can execute.
- Every Jenkins slave node also needs a directory on the slave machine for the slave agent to run build jobs, which can be defined in **Remote FS** root field.
- Labels are useful when the distributed build architecture is complex.
- You can define labels or tags to each build node and then configure a build job to run only on a slave node with a particular label.
- The Usage field lets you configure the usage of slave by Jenkins.
- In the **Launch Method** field, you can choose the strategy used to start a slave machine.
- The **Tools Location** under **Node Properties** helps you integrate different JDKs with your node.

Configuring Slave Machine with SSH

Screenshot of the newly created slave node is shown below:



Build Queue

No builds in the queue.

Build Executor Status

#	Master
1	Idle
2	Idle
	Slave 1
1	Idle
2	Idle

S	Name ↓	Response Time	Free Swap Space	Free Disk Space	Free Temp Space	Architecture	Clock Difference
	master	1ms	N/A	122GB	122GB	Mac OS X (x86_64)	In sync
	Slave 1	N/A	N/A	N/A	N/A		N/A

Starting Slave Machine with Java Web Start

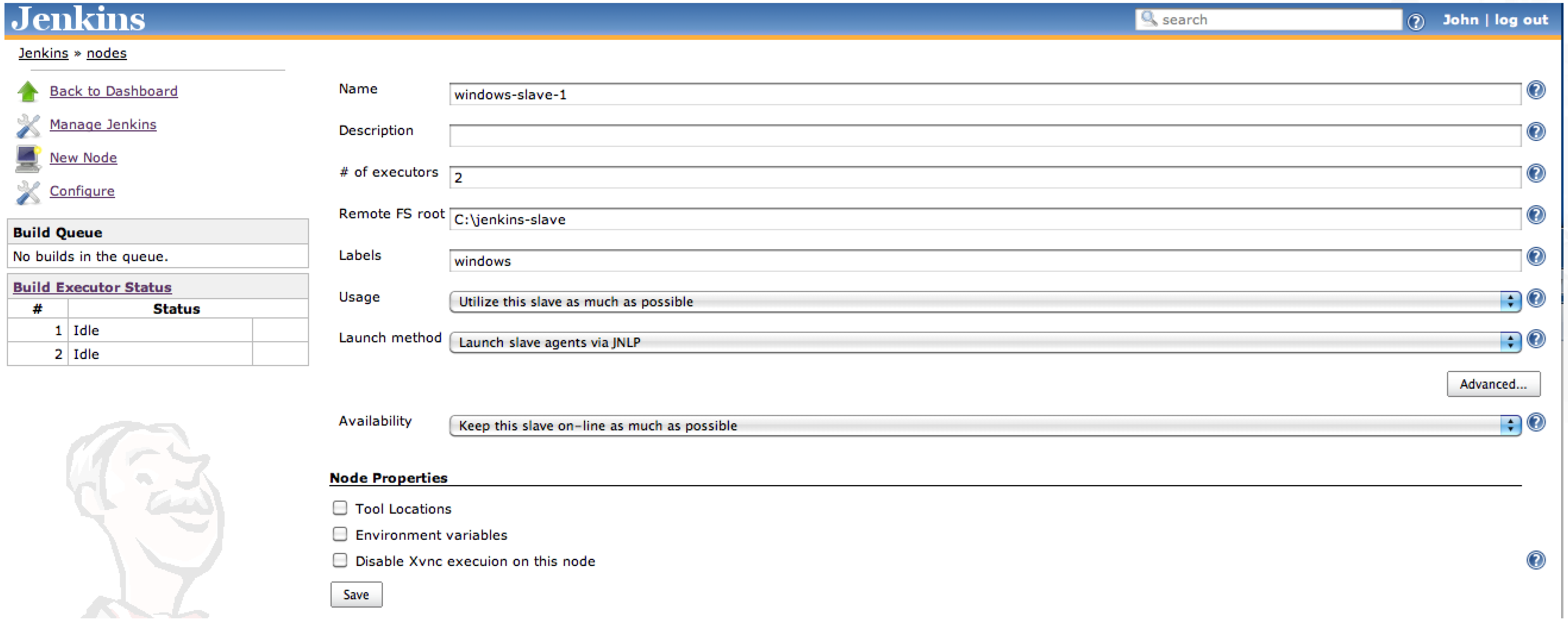
Java Web Start is used to connect the server to the slave. For example: If the slave machine is running on another network.

- It works no matter what operating system the slave is running on.
- It is more commonly used for slaves on windows machine.
- However, the limitation of this approach is that slave node cannot be started or restarted automatically by Jenkins.

Configuring Slave Machine with Java Web Start

- Steps to start a slave node with JNLP are:

1. Create a new slave node.
2. Select **Launch slave agents via JNLP** option in **Launch Method** field.
3. If it is a Windows slave node, provide a Windows path to Remote FS.


Jenkins

search | John | log out

Jenkins » nodes

Back to Dashboard
Manage Jenkins
New Node
Configure

Build Queue
No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

Name	windows-slave-1
Description	
# of executors	2
Remote FS root	C:\jenkins-slave
Labels	windows
Usage	Utilize this slave as much as possible
Launch method	Launch slave agents via JNLP
Availability	Keep this slave on-line as much as possible

Advanced...

Node Properties

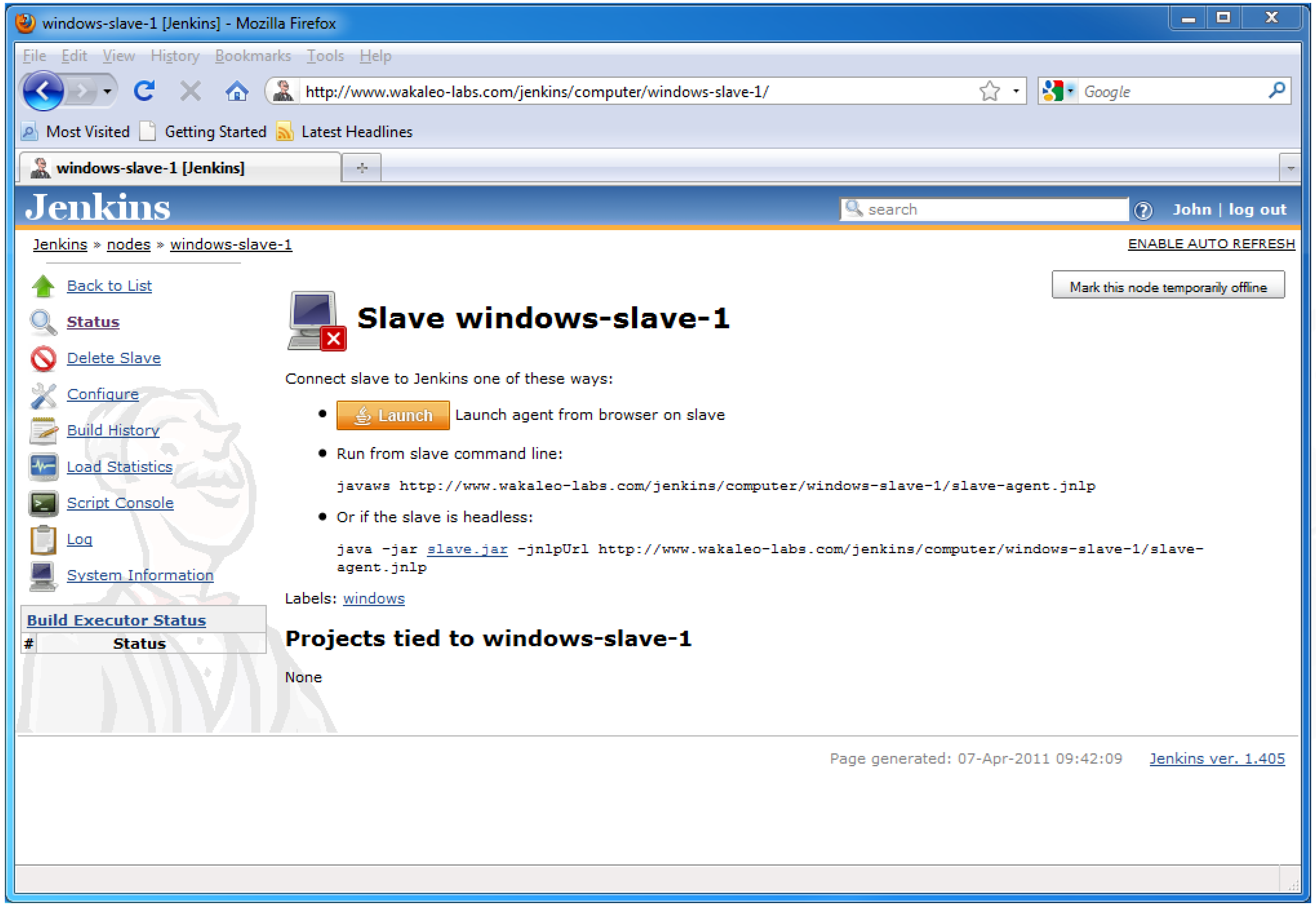
- [ ] Tool Locations
- [ ] Environment variables
- [ ] Disable Xvnc execuion on this node

Save

Configuring Slave Machine with Java Web Start

- Once you have configured slave node, log on to the slave machine and open the slave node browser screen as shown here:

- Click on the orange **Launch** button to start the node.


windows-slave-1 [Jenkins] - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.wakaleo-labs.com/jenkins/computer/windows-slave-1/
Google
Most Visited
Getting Started
Latest Headlines
windows-slave-1 [Jenkins]
Jenkins
search
John | log out
Jenkins » nodes » windows-slave-1
ENABLE AUTO REFRESH
Back to List
Status
Delete Slave
Configure
Build History
Load Statistics
Script Console
Log
System Information
Build Executor Status
#
Status
Mark this node temporarily offline
Launch

Slave windows-slave-1

Connect slave to Jenkins one of these ways:

- Launch: Launch agent from browser on slave
- Run from slave command line:

```
javaws http://www.wakaleo-labs.com/jenkins/computer/windows-slave-1/slave-agent.jnlp
```

- Or if the slave is headless:

```
java -jar slave.jar -jnlpUrl http://www.wakaleo-labs.com/jenkins/computer/windows-slave-1/slave-
agent.jnlp
```

Labels: windows

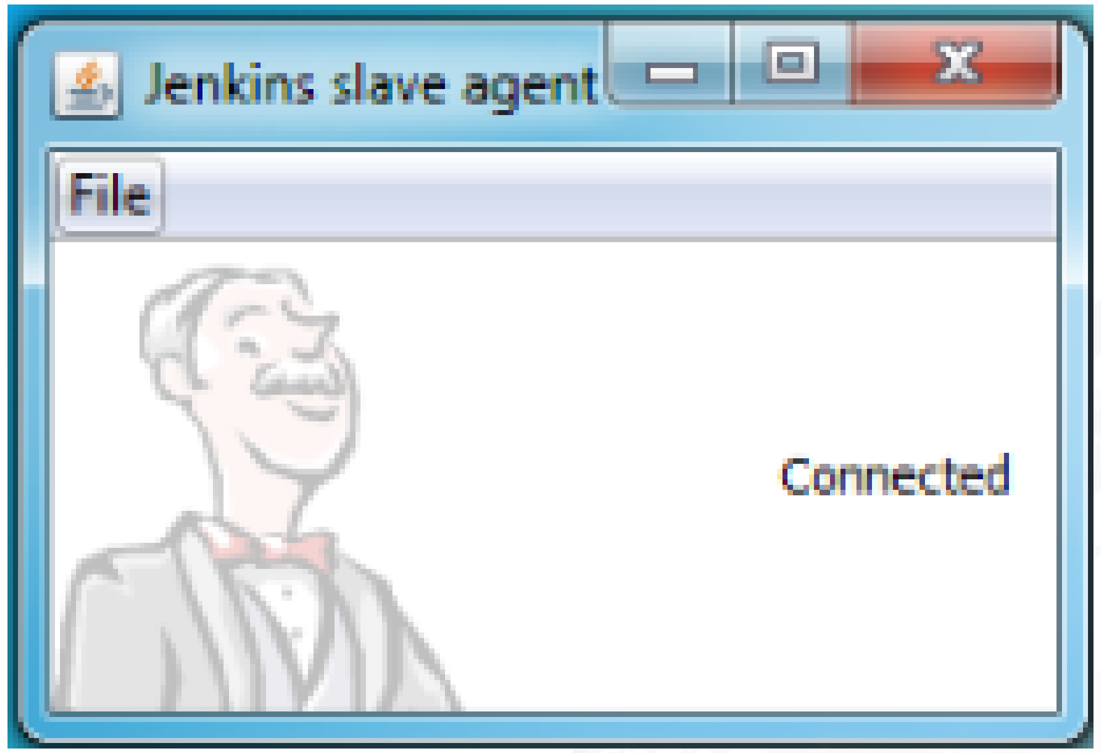
Projects tied to windows-slave-1

None

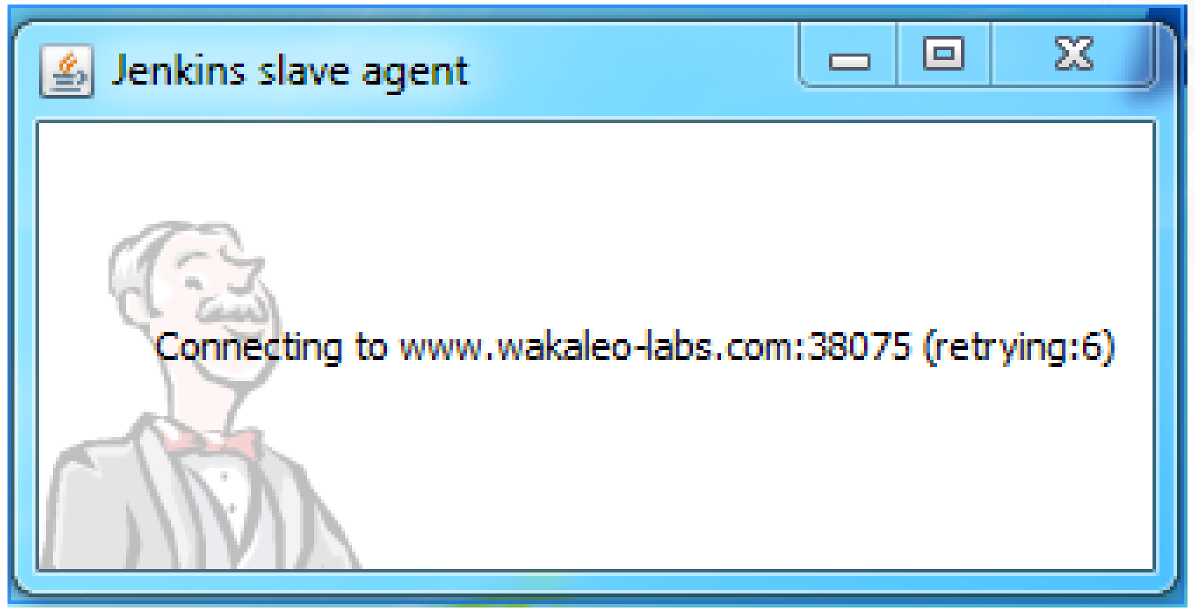
Page generated: 07-Apr-2011 09:42:09 Jenkins ver. 1.405

Configuring Slave Machine with Java Web Start

- The slave node is opened in a new window as shown in the figure:
- It is recommended to use Firefox or Chrome with Java Web Start as Internet Explorer may require additional browser level configurations.
- If security is activated on the Jenkins server, Jenkins will communicate with the slave on a specific nonstandard port.
- If there is any issue with the port, slave won't open and it prompts the following message on the screen:



Connected

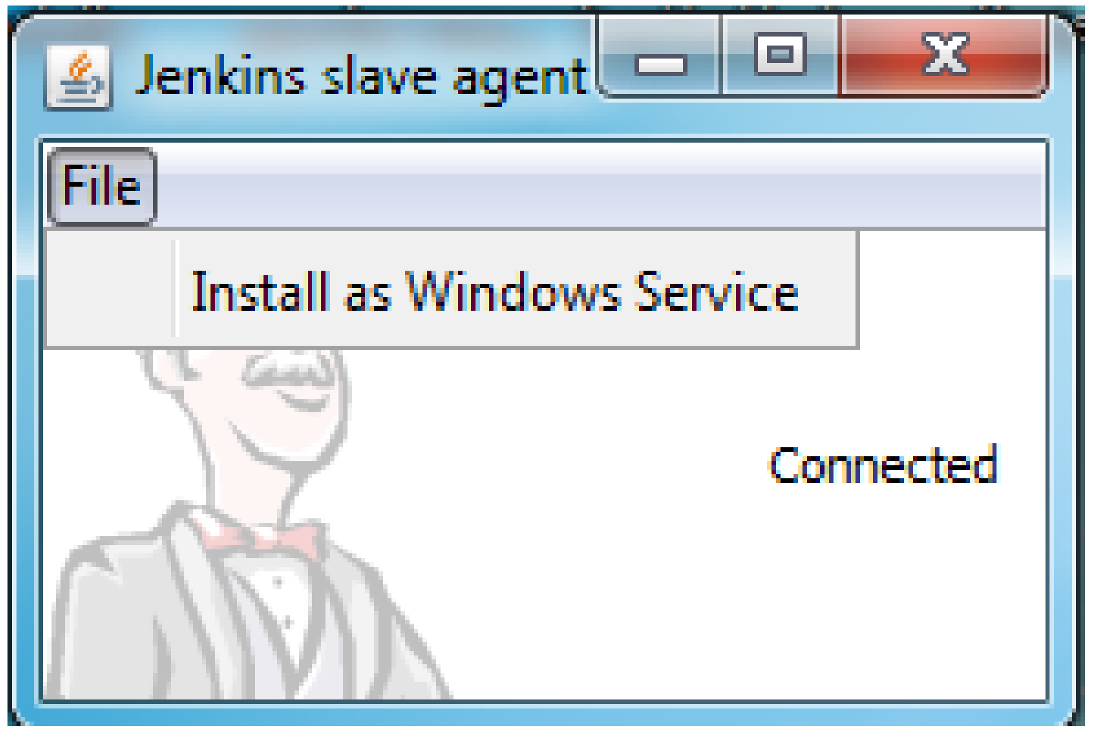


Connecting to www.wakaleo-labs.com:38075 (retrying:6)

Installing a Jenkins Slave as a Windows Service

Once the slave is up and running on the Windows machine, you can also install it as a Windows service by selecting the **Install as Windows Service** menu option in the File menu of the slave agent window

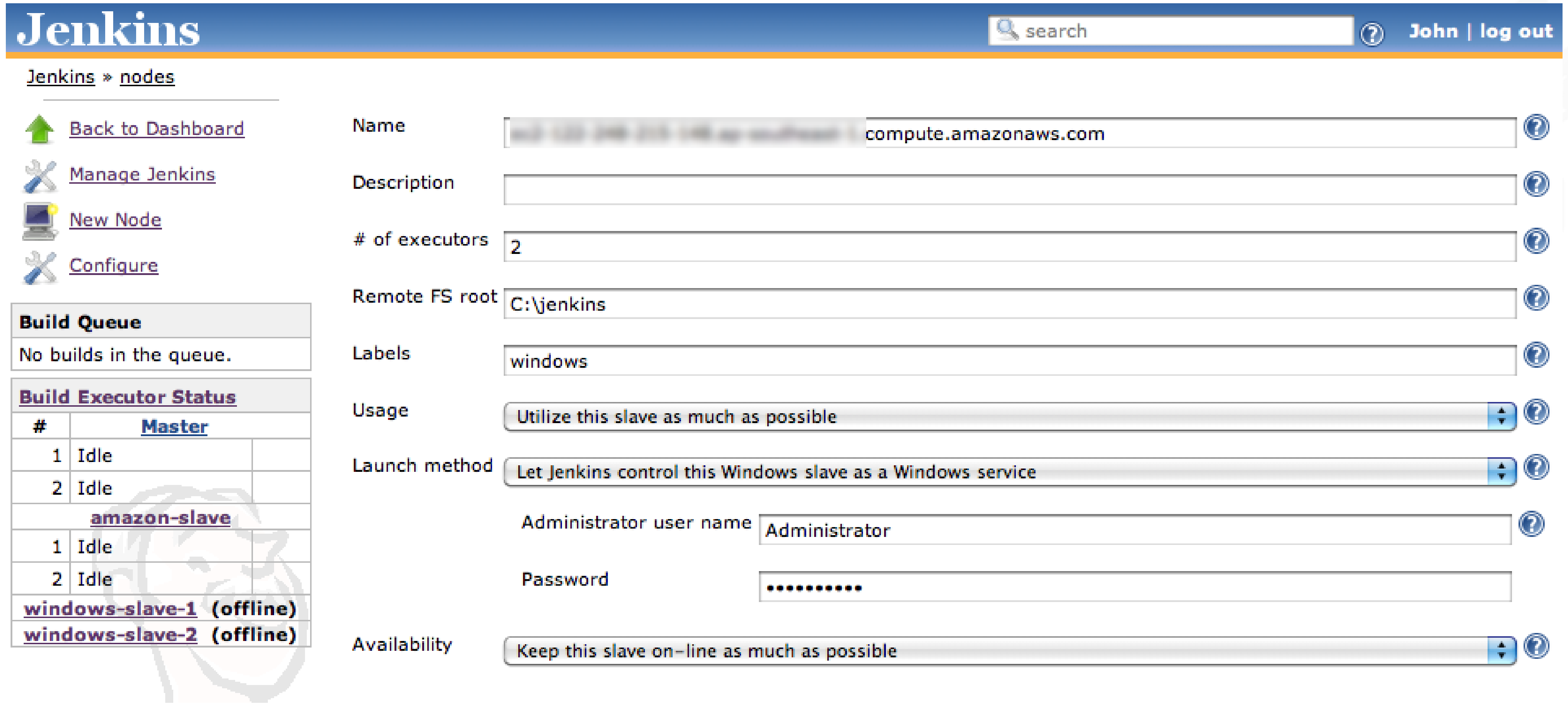
- Jenkins slave node will start automatically whenever the machine starts up after the installation.



Starting a Windows Slave as a Remote Service

Jenkins can be also used to control and operate remotely on the slave machine.

- While creating a node, choose **Let Jenkins control this Windows slave as a Windows service** and provide the username and password to the slave machine.
- Make sure that the username of the slave machine is same as the hostname of the slave machine.



Build Queue

No builds in the queue.

Build Executor Status

#	Master
1	Idle
2	Idle
	amazon-slave
1	Idle
2	Idle

windows-slave-1 (offline)

windows-slave-2 (offline)

Name	[redacted] compute.amazonaws.com
Description	
# of executors	2
Remote FS root	C:\jenkins
Labels	windows
Usage	Utilize this slave as much as possible
Launch method	Let Jenkins control this Windows slave as a Windows service
Administrator user name	Administrator
Password	••••••••••
Availability	Keep this slave on-line as much as possible

Associating a Build Job with a Slave or Group of Slaves

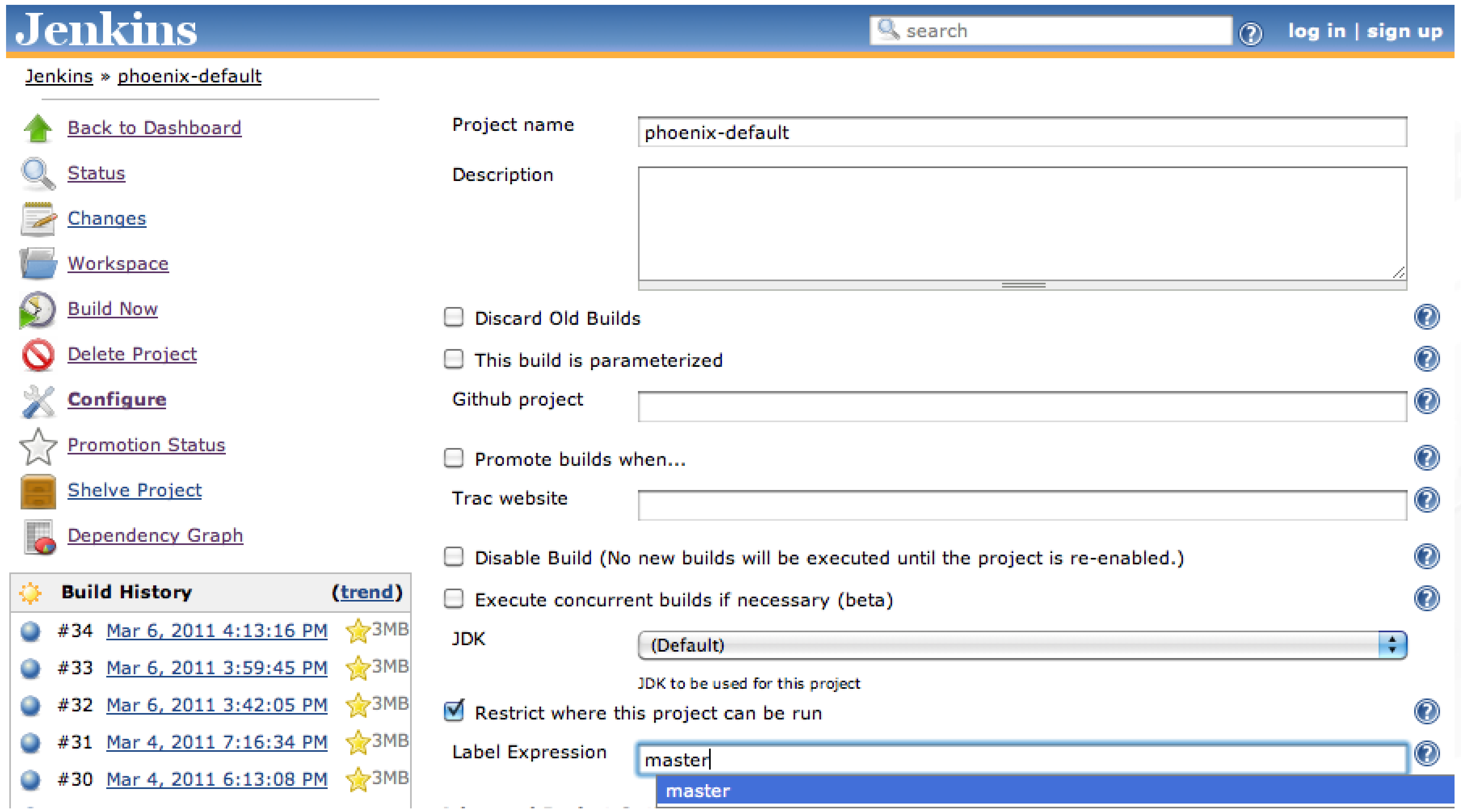
By default, Jenkins uses the first available slave node for optimal overall turn-around time. However, you can also configure Jenkins to use a particular slave machine to build jobs.

Follow the steps below:

1. Select the **Restrict where this project can be run** checkbox in the build configuration page.

1. Enter the name of the machine or a label of a group of machines in the **Label Expression** field.

1. Select the machine from the dropdown list.



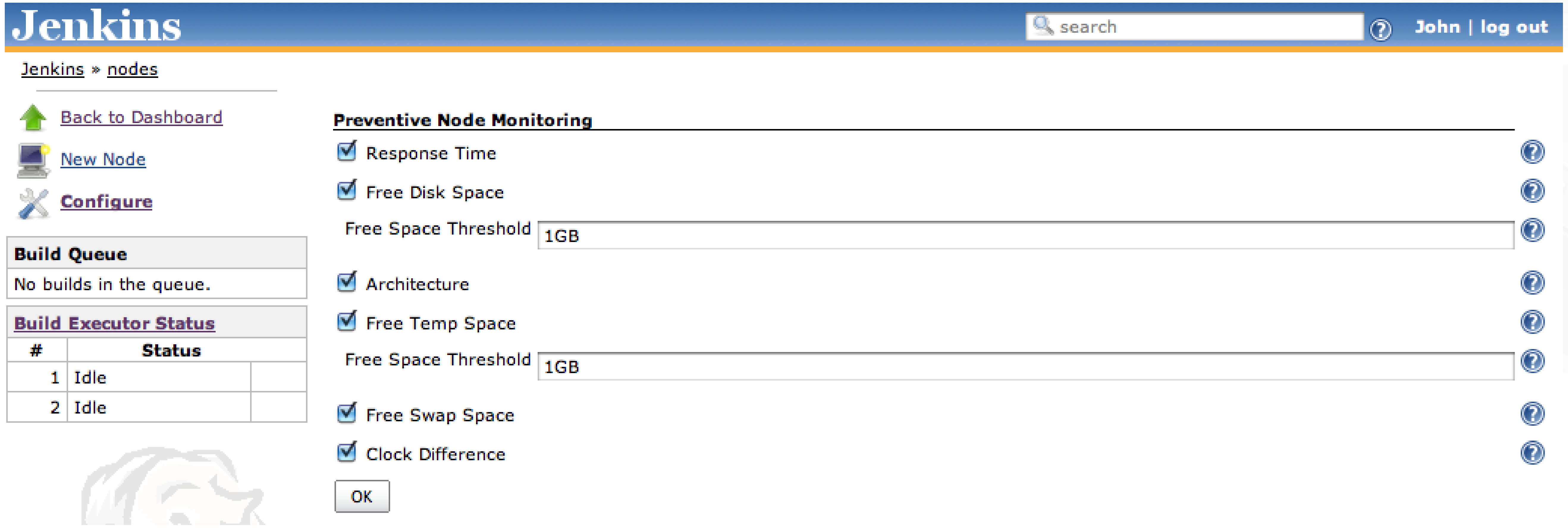
#34	Mar 6, 2011 4:13:16 PM	3MB
#33	Mar 6, 2011 3:59:45 PM	3MB
#32	Mar 6, 2011 3:42:05 PM	3MB
#31	Mar 4, 2011 7:16:34 PM	3MB
#30	Mar 4, 2011 6:13:08 PM	3MB

Project name: phoenix-default
Description:
☐ Discard Old Builds
☐ This build is parameterized
Github project:
☐ Promote builds when...
Trac website:
☐ Disable Build (No new builds will be executed until the project is re-enabled.)
☐ Execute concurrent builds if necessary (beta)
JDK: (Default)
JDK to be used for this project
☑ Restrict where this project can be run
Label Expression: master
master

Node Monitoring

The different ways Jenkins monitors the slave agents are:

1. Response time: A slow response time can indicate either a network problem or that the slave machine is down.
2. Disk space: Temporary directory space and swap space are available to the Jenkins user on the slave machine based on the complexity and agenda of build jobs.
3. System clocks synchronization: If the clocks are not correctly synchronized, errors can occur and incorrect reports may be generated.



Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

Preventive Node Monitoring

- ✓ Response Time
- ✓ Free Disk Space
  - Free Space Threshold: 1GB
- ✓ Architecture
- ✓ Free Temp Space
  - Free Space Threshold: 1GB
- ✓ Free Swap Space
- ✓ Clock Difference

OK

Build Jobs with Slave Groups

Duration: 30 mins.

Problem Statement: You are given a project to build jobs on a slave node in Jenkins.

Steps to perform:

1. Create slave nodes.
2. Log in to master node.
3. Create a new freestyle job.
4. In the configuration page, check the *Restrict where this project can be run* option and enter the name of the slave node.
5. Build job with slave machine.

ASSISTED PRACTICE

Key Takeaways

- Jenkins has the capability to execute and dispatch build jobs over a large number of systems.
- The number of build servers required can vary during the project life cycle.
- The role of Jenkins master job is to schedule build jobs, assign builds to the slaves, record and generate build results.
- Labels are tags to a slave node that allow node categorization in Jenkins.

Distributed Builds

Duration: 30 mins.

Problem Statement:
You're a DevOps engineer at PiperChat, a video chat company that started out as an internal communication tool. PiperChat has since expanded as a popular video calling app with multiple features like various image and voice filters. The company wants to make the build more efficient since the stack has grown. You're required to configure a master-slave Jenkins architecture to reduce the load on the single Jenkins instance. The company is also considering moving to the Cloud and wants to use this as a pilot. You're asked to set up a Jenkins slave node on an AWS EC2 machine while retaining the on-prem Jenkins instance as the master.

Requirements:
- The app should be built with Maven.
- The architecture should have one slave node on an AWS EC2 machine and a local master node.
- The master node should only manage slave nodes and all build jobs must be run on the slaves.

Building a CI/CD Pipeline with Jenkin

Duration: 60 min.

Project Objective:

Use Jenkins to set up a CI/CD pipeline that will compile and test a Maven project and deploy it to a Tomcat Server.

COURSE-END PROJECT

simplilearn

Background of the Project Statement

You’re a DevOps engineer at Pied Piper, a software company that provides web and mobile app development services. Your team is tasked with building a web app for an online book shop named Bookzy. The developers have decided to use Springboot to generate the project, Maven for compilation, and Git as the Source Code Management system for the project.You’re required to set up a Jenkins pipeline that involves three different freestyle jobs for code compilation, testing, and deployment respectively. Your Pipeline has to poll the SCM nightly for commits, build the project, and trigger the test job if the build is stable. The test job has to run unit tests, publish the JUnit test report, and trigger the freestyle job for deployment. The deployment job is required to package the Maven project, deploy the war file to a Tomcat server, and notify you via email if deployment failed.

You Are Asked to Do

- ❖ Build the app with Spring Boot and Maven
- ❖ Set up the Tomcat server on an EC2 instance
- ❖ Configure the Tomcat server to allow remote deployment
- ❖ Test and publish the results with JUnit
- ❖ Build a pipeline with three freestyle jobs for :
  - ➢ Compilation
  - ➢ Test
  - ➢ Deployment

You must use the following

Jenkins

Programming language: Java

Git and GitHub

Spring Boot

AWS EC2

Tomcat

Architecting Jenkins Pipeline for Scale

Duration: 60 min.

Project Objective:

Use Jenkins to set up a distributed pipeline that will compile and test a Maven project on two different slave nodes respectively.

COURSE-END PROJECT

Background of the Project Statement

You’re a DevOps engineer at Softmax Solutions, a software company that develops image filters for various photo enhancement apps. It is undergoing an infrastructural change to implement DevOps in its development process. The company uses git as their Source Code Management System and AWS for hosting its servers. You’re required to architect a scalable Jenkins Pipeline for building and testing the software stack. You’re tasked with designing a Jenkins architecture that involves one master and two slave nodes, all hosted on various AWS instances. The build jobs should always be triggered by the master and executed on the slaves. You have to set up a pipeline on the master node and write a Groovy script that clearly differentiates the tasks to be run on various slaves.

You Are Asked to Do

- ❖ Build the app with Spring Boot and Maven
- ❖ Set up three EC2 instances to host the Jenkins architecture
- ❖ Configure a master node to trigger and monitor all builds
- ❖ Set up dedicated slave nodes for testing and compiling
- ❖ Define a Jenkinsfile that marks each node and their functions
- ❖ Build a pipeline with three stages:
    - ➢ Checkout
    - ➢ Compilation
    - ➢ Test

You must use the following

Jenkins

Programming language: Java

Git and GitHub

Spring Boot

AWS EC2

Maven