git ---->

burocracy----> step by step provccess (ex: gvt system)

blue green deplyement ---- it is with out down time deployement

dns name-

swap

whitelist ip add cheyali automation (task)

192.0.2.0/32

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

From Me to Everyone: 03:05 PM

hey

From Shantanu Patra to Everyone: 03:15 P

in case of cluster environment - you can search all of your data all the time does not matter your server is down or up, if any of the srver is up you can search all of your data.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

sudo apt-get update

sudo apt-get install grafana

cd usr /bin

ufw status

ufw enable

ufw allow

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

hands on lab jenkins task1:-

jenkins installation:-redhat /centos

1.install java-1.8.0-openjdk-devel

sudo yum install -y java-1.8.0-openjdk-devel

1.Install wget:

sudo yum install -y wget

2.Download the repo:

sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenkins.repo

3.Import the required key:

sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key

4.Install Jenkins:

sudo yum install -y jenkins

II.Building in Jenkins /task2

a)create a folder named test:

In the dashboard click on create item, from there enter "Test" at the top of the page and then click "folder" and then "ok"

b)create a user test job that saves it output in afile "user\_test.txt" :

in the folder you created, go to new item on the left, then name the item user\_test and then click "freestyle build" and then "ok" In the job configuration page in the build step, click execute shell, then in the text box type uname -a && whoami > user\_test.txt then click save.

Then click build on the left, Then click workspace and locate the file, view the contets of the file by clicking view on the page. This should result in the username 'jenkins'

iii.Building from scm/task2:

guide:

Building from SCM

Introduction

In this lab we configure Maven to perform a build. This includes pulling the source code for the build from SCM. At the end of the build process, we also create an artifact for the build.

Solution

Configure Maven Installer

1.a browser to navigate to the provided public IP address for the server. Remember to add ":8080" to the end of the IP address to specify the default Jenkins port.

2.Log in with the credentials provided in the lab guide instructions.

3.Click Manage Jenkins.

4.Click Global Tool Configuration.

5.Under Maven installations, click Add Maven.

6.In the Name box, enter "M3".

7.Make sure Install automatically is checked.

8.Click Save.

Configure the Build to Use Maven and Make the Index File

1.Click New Item.

2.Enter an item name of "mavenproject" in the box provided.

3.Select Freestyle project.

4.Click OK.

5.Click the Source Code Management tab at the top of the screen.

6.Select the option for a Git repository.

7.Copy the git repository link from the lab instructions and enter it into the Repository URL box.

8.Click the Build tab at the top of the screen.

9.Click Add build step and select the Invoke top-level Maven targets option.

10.Under Maven Version, select M3.

11.In the Goals box, enter "clean package".

12.Click Add build step and select the Execute shell option.

13.In the Command window, enter "bin/makeindex".

14.Click Add post-build action and select the Archive the artifacts option.

15.Inside the Archive the artifacts box, click Advanced...

16.Check the option for Fingerprint all archived artifacts.

17.In the Files to archive box, enter "index.jsp".

18.Click Save.

19.Click Build Now.

20.Refresh the window and click the View link next to index.jsp. Verify the contents of the index.jsp file.

The URI of the git repository for this lesson is https://github.com/linuxacademy/content-cje-prebuild.git

gitclone https://github.com/linuxacademy/content-cje-prebuild.git

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

iv.Distributing a Build/task4 :

1.configure the slave machine for use with the jenkins master

i.Modify the /etc/passwd file on the master to allow the Jenkins user a shell login.

ii.Add a password to the jenkins user so you are able to su to that user.

iii.Once this is done, su to the jenkins user and run ssh-keygen.

4.Then, as the jenkins user, ssh to the slave machine.

5.Create a jenkins user, add them to sudoers, and create the entry with nopasswd.

6.Exchange the jenkins user's ssh keys with the slave server.

7.Verify key login is working on the slave server.

2.run the maven bulid on the remote Agent:-

1.Create a Maven installation named M3.

2.From Manage Jenkins > Manage Nodes, set up the remote node.

3.Label the node slave1.

4.Create a freestyle project named mavenproject that is set to run only on slave1.

5.Configure it to use M3 maven install.

6.Create a shell build step to run bin/makeindex.

7.Archive the index.jsp and ensure it gets fingerprinted

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

V.Working with Artifacts and Fingerprints: task5:

a.create the 'webdev' folder and the 'index'project:

1.Once logged in to the Jenkins Dashboard, configure Maven on the server and name it M3.

2.Under new item, create a folder named webdev.

3.Inside the webdev folder, make a freestyle project named index.

4.Configure the project using the Git repo in the instructions, and ensure the M3 Maven Server is configured.

5.Run the shell command bin/makeindex, and then archive the index.jsp file.

b.install the copy artifact plugin :

1.Navigate to the main page Jenkins > Manage Jenkins > Manage Plugins.

2.Click the Available tab.

3.Search for copy artifact in the upper-right Filter field.

4.Click the radio button for Copy Artifact.

5.Click Download now and install after restart.

6.Click Restart Jenkins when installation is complete and no jobs are running.

7.Wait for Jenkins to restart.

check\_circle

c.

create and bulid the 'tomcat' project:

1.Inside the backend > tomcat folder, create a freestyle project named tomcat.

2.The first build step should be copying the artifact from the webdev/index job to src/main/webapp in the tomcat job.

3.Using the M3 Maven Server, run clean package.

4.Create a post-build step that published JUnit tests from the target/surefire-reports/\*.xml directory.

5Run the build and confirm that the tests are published.

vi.Security in Jenkins : task6:

a. enable project based matrix security:

go to manage jenkins > global security and enable project based matrix seecurity. Ensure that authenticated users have global read access at this level.

b.create users,james,diane and laura:

Go to manage jenkins > manage users and create accounts for James, Diane and Laura

Remember the passwords so that you can Audit secrity as you set it.

c.create webdev folder and give james access:

Create a top level folder named webdev.

Enable project security and disallow inheritance

Add james and give him full access.

Make a freestyle project named test, and set it to inherit from parent.

d.create backend folder and give diane full access:

Create a top level folder named backend, in the folder config enable project based security

Disable inheritance and add Diane and grant full access

e.create tomcat folder inside the backend folder and laura access to this folder.

verify that james is not able to access the backend folder ,diane :

Go into the backend folde, create a new item, create a folder named tomcat.

On this folder configure project based security and add laura with full access and alllow inheritance from parent.

Make sure that on the backend folder Laura has permissons to view job so that she can see the tomcat folder.

inside tomcat make a test freestyle project.

f. cannot see the webdev folder but can see tomcat laura can only acess tomcat:

Login as each user and attempt to access the folders that they should not be allowed access to.

vii.Parameterizing builds /task7:

i.create the webdev folder and index freestyleproject:

On the dash board create a new item named webdev make it a folder.

Inside of webdev make a freestyle project named index.

ii.configure the index project to accept the name steve:

In the index project click this build requires parameters.

Add a string parameter called name, the value should be steve

Add the git repo

add the maven server M3

add the shell command to make the bin/makeindex

Make a post build action to archive the artifact.

check the artifat to ensure that it says hello steve

Viii.Bulid Triggers:

i.create webdev/index and archive index.jsp

Configure maven M3

Create webdev folder

Configure the job to take parameters: name=steve

Pull from Git using the URL that is in the instructions

Build the top-level maven 'clean package'

In a shell, run bin/makeindex

Archive index.jsp.

ii.create backend/tomcat and trigger it to bulid on index complete >sucessful:

Create the backend folder and tomcat subfolder.

Create the tomcat frestyle project.

Configure trigger on index successful

Pull from the Git Url in lab instructions.

Build top-level maven 'clean package'

Make sure that it runs after index completes.

Check index for steve.

0