Nexus Setup

Raghuram PSS

Pre-requisites:	Docker-Engine Ports 8081, 18078 enabled
Create a folder "nexus-data" under root home Give full permissions	cd mkdir nexus-data chmod 777 nexus-data
Pull nexus3 image	docker pull sonatype/nexus
Start the nexus container	docker run --name nexus --restart=always -d -p 8082:8081 -p 18078:18078 -v $(pwd):/nexus-data sonatype/nexus
Access the nexus repository	<IP>:8081/nexus
Login to the nexus repository	Username: admin; password: admin123
Create a new repository	Settings -> Repositories -> Create Repository • Pick "Hosted Repository" • Repository Id: NexusRepo • Repository Name: NexusRepo • Repository Type: Hosted • Provider: Maven2 • Save

Deploy to Nexus

Pre-requisites	VM with Ubuntu 16.04 Docker-Engine
Create a folder for Jenkins	mkdir myjen chmod 777 myjen cd myjen
Start Jenkins	docker run --name myjen -d -p 8082:8080 -v $(pwd):/var/jenkins_home jenkins
Add Plugin	Configure Plugins -> Nexus Artifact Uploader; Nexus Platform Plugin
Install Maven	Manage Jenkins-> Global Tool Configuration -> Maven • Click "Maven Installations" • Name: M3 • Check "Install Automatically" • Install From Apache • Version 3.5.2
	1. Select Manage Jenkins from the Dashboard's left-navigation menu. 2. Select Configure System from the list of configuration options. 3. In the Sonatype Nexus section, click the Add Nexus Repository Manager Server dropdown menu and then select Nexus Repository Manager 2.x Server. Enter the following: • Display Name: Name of the server you want shown when selecting Nexus Repository Manager instances for build jobs. ("**NexusRepo**") • Server ID: A unique ID used to reference Nexus Repository Manager in Build Pipeline scripts. It should be alphanumeric without spaces. ("**Nexus123**") • Server URL: Location of your Nexus Repository Manager server. (http://35.154.11.9:8081/nexus) • Credentials: Select the Add button to enter your Nexus Repository Manager username and password using the Jenkins Provider Credentials: Jenkins modal window. Once added, select your Nexus Repository Manager username and password from the Credentials dropdown list. 4. Click the Test Connection button. 5. After a successful connection to Nexus Repository Manager, click the Save button
Git - Pom.xml	<distributionmanagement> <repository>
Create Jenkins job	1. Create freestyle project "nexus_test" 2. Source Code Management -> Git -> Repository URL -> https://github.com/raghupss/Hellow_world.git 3. In the Build section of the configuration screen, click the Add Build Step… 1. Add a build step a. Invoke top-level Maven targets i. Maven Version : M3 ii. Goals: package 2. Add a build step Select Nexus Repository Manager Publisher. Enter the following parameters: a. Nexus Instance: Select the display name set in global configuration. (NexusRepo) b. Nexus Repository: Select a repository that has release repository policy and allows for artifact uploads. (hello_maven) c. Packages: Select packages to publish to Nexus Repository Manager during your freestyle build. For this example, use the Add Package dropdown to select a Maven Package. d. Group -> com.efsavage e. Artifcat -> hello-world-war f. Version -> 1.0.0 g. Packaging -> war h. Artifacts i. File Path -> target/hello-world-war-1.0.0.war 4. Apply and Save.

Chef pipeline e.g.

Step-1	Create a cookbook to create folder "appfolder"	chef generate cookbook appfolder

		directory '/etc/appfolder' do
		  owner 'root'
		  group 'root'
		  mode '0755'
		  action :create
		end

	Test it locally	chef-client --local-mode --runlist 'recipe[appfolder]'
	Add to run-list	knife node run_list add ws-1 appfolder
Step-2	Create a cookbook "deploy_war" to push .war to node	chef generate cookbook deploy_war

		remote_file "/etc/appfolder/hello-world-war-1.0.0.war" do
		  source "http://13.127.133.123:8082/nexus/content/repositories/NexusRepo/com/efsavage/hello-world-war/1.0.0/"
		  mode '0755'
		  owner "root"
		  group 'root'
		  #notifies :restart, "service[tomcat8]"
		end

	Test it locally	chef-client --local-mode --runlist 'recipe[deploy_war]'
	Add to run-list	knife node run_list add webserver-1 deploy_war
Step 3	Create a cookbook to run tomcat container	chef generate cookbook docker_tomcat
	Place a dependency on the docker-compose cookbook in your cookbook's metadata.rb	nano metadata.rb
		depends 'docker-engine', '~> 0.2.3' #depends 'docker_compose', '~> 0.1.1'

	Create files/docker-compose_tomcat.yml	web-server:
		    image: tomcat:8.0
		    ports:
		      - "8080:8080"
		    volumes:
		      - /etc/appfolder/:/usr/local/tomcat/webapps/

	In the recipes/default.rb	

	Include	include_recipe 'docker_compose::installation'
		
		# Provision Compose file
		cookbook_file '/etc/docker-compose_tomcat.yml' do
		  source 'docker-compose_tomcat.yml'
		  owner 'root'
		  group 'root'
		  mode 0640
		  notifies :up, 'docker_compose_application[tomcat]', :delayed
		end
		
		# Provision Compose application
		docker_compose_application 'tomcat' do
		  action :up
		  compose_files ['/etc/docker-compose_tomcat.yml']
		end

		knife node run_list add webserver-1 docker_tomcat
	Run berks install from docker_tomcat coobkook	berks install
	Then Upload all the cookbooks	berks upload
	Goto chef server and add all cookbooks to the node(s)	
	Goto the node and run chef-client	