26.05.2022

SETTING UP JENKINS PIPELINE TO DEPLOY DOCKER SWARM PROJECT

Prepared by: PRIYANKA DAS

Project Objective :

You have to develop an environment for Docker networking.

Use The Following :

- Jenkins: To create a pipeline to deploy Docker Swarm
- Docker Swarm: To implement container networking
- Git: To connect and push files from the local system to GitHub
- GitHub: To store the Angular application

Background of the problem statement:

As you have worked on Docker containers previously, your manager has asked you to perform container scheduling over multiple hosts using Docker CLI and connect multiple hosts with Docker containers

Following Requirements Should Be Met :

- A few of the source code should be tracked on GitHub repositories. You need to document the tracked files that are ignored during the final push to the GitHub repository.

- Submission of your GitHub repository link is mandatory. In order to track your task, you need to share the link of the repository in the document.

- The step-by-step process involved in completing this task should be documented

Step by step process :

The assumption here is that Vagrant, VirtualBox and Gitbash are already install on your machine (my development environment was a Windows 10 Pro machine)
Execute the following commands, in gitbash, in order to create a two-node docker swarm mode cluster. The nodes are based on 'ubuntu/xenial64' VM. Once the cluster is created successfully, log in to the master node:

- git clone https://github.com/shazChaudhry/docker-swarm-mode.git
- cd docker-swarm-mode
- vagrant up
- vagrant ssh node1 (Log in to the master node)
- docker node ls (confirm that there are two nodes in the cluster; master and worker)
- cd /vagrant

Deploy stack by running the following commands which will utilize Docker secrets for Jenkins and proxy.

- Jenkins secrets are defined in the "./secrets/jenkins" directory
- Proxy's secrets are defined in the "./certs" directory

docker stack deploy --compose-file docker-compose.portainer.yml portainer
docker stack deploy --compose-file docker-compose.yml ci

Service URLs:

- http://node1:9000 (Portainer)
- https://node1/jenkins (Jenkins). admin username: admin; Password: admin
- https://node1/sonar (SonarQube). admin username: admin; Password: admin
- https://node1/nexus (Nexus). admin username: admin; Password: admin123
  - Follow these instructions to setup docker registries; at least one should be of type proxy and should point to docker hub: https://help.sonatype.com/repomanager3/private-registry-for-docker/proxy-repository-for-docker
  - Reserve "8082" for docker group repo
  - Reserve "5000" docker hosted repo
- https://node1/gitlab (Gitlab CE). admin username: root; Password: Password01
  - Gitlab takes a few minutes to become available so please be a little patient :)
  - You can also view service logs, docker service logs -f ci_gitlab, to check when gitlab becomes available.

Deploy CI stack on "Docker for AWS" :

It is assumed you have followed Docker for AWS documentation to create a new VPC. Follow these commands in an ssh client to log in to your master node (I'm using gitbash on Windows 10 Pro).
Please note you can not ssh directly into worker nodes. You have to use a manager node as a jump box

- eval $(ssh-agent) OR exec ssh-agent bash
- ssh-add -k ~/.ssh/personal.pem (You wiill have to use your own key)
- ssh-add -L
- ssh -A docker@<Manager Public IP>
- cat /etc/*-release
- docker node ls

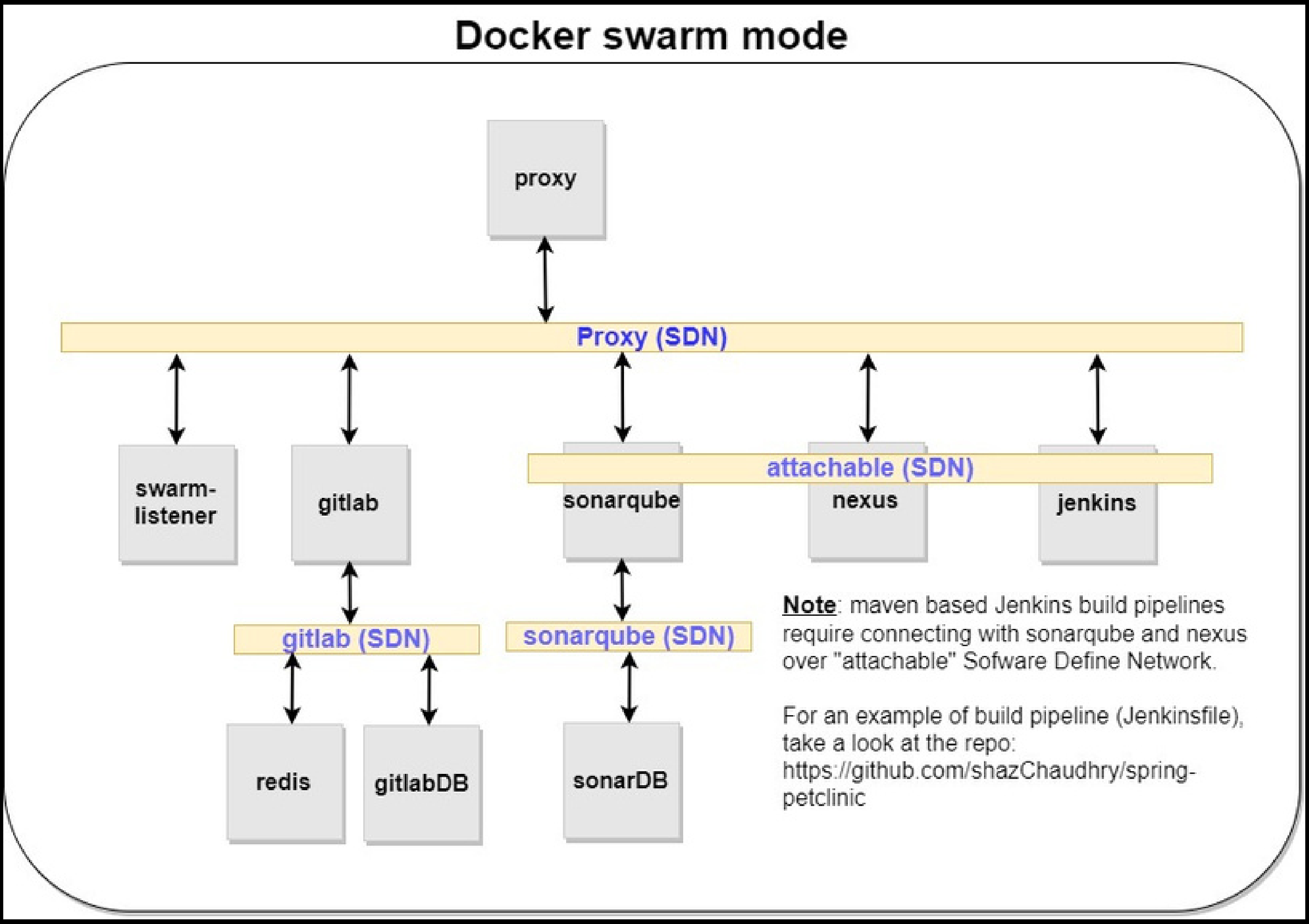
Clone this repo and change directory by following these commands:

- alias git='docker run -it --rm --name git -v $PWD:/git -w /git alpine/git' (This alias is only required if git is not already installed on your machine. This alias will allow you to clone the repo using a git container)
- git version
- git clone https://github.com/shazChaudhry/docker-swarm-mode.git
- sudo chown -R $USER:$USER docker-swarm-mode
- cd docker-swarm-mode

Start the Portainer by running:

- docker stack deploy -c docker-compose.portainer.yml portainer

Flowchart :



Note: maven based Jenkins build pipelines require connecting with sonarqube and nexus over "attachable" Sofware Define Network.

For an example of build pipeline (Jenkinsfile), take a look at the repo: https://github.com/shazChaudhry/spring-petclinic