|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> |
|  | <settings xmlns="http://maven.apache.org/SETTINGS/1.0.0" |
|  | xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" |
|  | xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 https://maven.apache.org/xsd/settings-1.0.0.xsd"> |
|  | <localRepository>${user.home}/.m2/repository</localRepository> |
|  | <pluginGroups> |
|  | <pluginGroup>org.sonarsource.scanner.maven</pluginGroup> |
|  | </pluginGroups> |
|  | <servers> |
|  | <server> |
|  | <id>maven-snapshots</id> |
|  | <username>admin</username> |
|  | <password>admin123</password> |
|  | </server> |
|  | <server> |
|  | <id>maven-releases</id> |
|  | <username>admin</username> |
|  | <password>admin123</password> |
|  | </server> |
|  | </servers> |
|  | <mirrors> |
|  | <mirror> |
|  | <!--This sends everything else to /public --> |
|  | <id>nexus</id> |
|  | <mirrorOf>external:\*</mirrorOf> |
|  | <!-- your address may differ: --> |
|  | <url>http://nexus:8081/nexus/repository/maven-public/</url> |
|  | </mirror> |
|  | </mirrors> |
|  | <profiles> |
|  | <profile> |
|  | <id>sonar</id> |
|  | <activation> |
|  | <activeByDefault>true</activeByDefault> |
|  | </activation> |
|  | <properties> |
|  | <!-- Optional URL to server. Default value is http://localhost:9000 --> |
|  | <sonar.host.url>http://sonarqube:9000/sonar</sonar.host.url> |
|  | </properties> |
|  | </profile> |
|  | <profile> |
|  | <id>nexus</id> |
|  | <!--Enable snapshots for the built in central repo to direct --> |
|  | <!--all requests to nexus via the mirror --> |
|  | <repositories> |
|  | <repository> |
|  | <id>central</id> |
|  | <url>http://central</url> |
|  | <releases><enabled>true</enabled></releases> |
|  | <snapshots><enabled>true</enabled></snapshots> |
|  | </repository> |
|  | </repositories> |
|  | <pluginRepositories> |
|  | <pluginRepository> |
|  | <id>central</id> |
|  | <url>http://central</url> |
|  | <releases><enabled>true</enabled></releases> |
|  | <snapshots><enabled>true</enabled></snapshots> |
|  | </pluginRepository> |
|  | </pluginRepositories> |
|  | </profile> |
|  | </profiles> |
|  | <activeProfiles> |
|  | <!--make the profile active all the time --> |
|  | <activeProfile>nexus</activeProfile> |
|  | </activeProfiles> |
|  | </settings> |

|  |
| --- |
| #!/bin/bash |
|  |  |
|  | docker info |
|  |  |
|  | docker stack deploy --compose-file docker-compose.yml ci |

# -\*- mode: ruby -\*-

# vi: set ft=ruby :

$docker\_swarm\_init = <<SCRIPT

docker swarm init --advertise-addr 192.168.99.101 --listen-addr 192.168.99.101:2377

docker swarm join-token --quiet worker > /vagrant/worker\_token

SCRIPT

Vagrant.configure("2") do |config|

# https://app.vagrantup.com/ralfkrause/boxes/centos7

config.vm.box = "ralfkrause/centos7"

config.hostmanager.enabled = true

config.hostmanager.manage\_host = true

config.hostmanager.manage\_guest = true

config.vm.provision "docker"

config.vm.define "node1", primary: true do |node1|

node1.vm.hostname = 'node1'

node1.vm.network :private\_network, ip: "192.168.99.101"

node1.vm.provider :virtualbox do |vb|

vb.customize ["modifyvm", :id, "--natdnshostresolver1", "on"]

vb.customize ["modifyvm", :id, "--memory", 8000]

vb.customize ["modifyvm", :id, "--name", "node1"]

end

node1.vm.provision :shell, inline: $docker\_swarm\_init

end

config.vm.define "node2" do |node2|

node2.vm.hostname = 'node2'

node2.vm.network :private\_network, ip: "192.168.99.102"

node2.vm.provider :virtualbox do |vb|

vb.customize ["modifyvm", :id, "--natdnshostresolver1", "on"]

vb.customize ["modifyvm", :id, "--memory", 2000]

vb.customize ["modifyvm", :id, "--name", "node2"]

end

node2.vm.provision :shell, inline: "docker swarm join --token $(cat /vagrant/worker\_token) 192.168.99.101:2377"

end

end

version: "3.7"

volumes:

gitlabPostgresql\_data:

driver: "cloudstor:aws"

driver\_opts:

ebstype: gp2 # https://docs.docker.com/docker-for-aws/persistent-data-volumes/#use-a-unique-volume-per-task-using-ebs && http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html

size: 25

iops: 1000

backing: relocatable

gitlab\_data:

driver: "cloudstor:aws"

driver\_opts:

ebstype: gp2

size: 25

iops: 1000

backing: relocatable

jenkins\_home:

driver: "cloudstor:aws"

driver\_opts:

ebstype: gp2

size: 25

iops: 1000

backing: relocatable

nexus\_data:

driver: "cloudstor:aws"

driver\_opts:

ebstype: gp2

size: 25

iops: 1000

backing: relocatable

postgresql:

postgresql\_data:

redis\_data:

sonarqube\_bundled\_plugins:

sonarqube\_conf:

sonarqube\_data:

sonarqube\_extensions:

secrets:

cert-xip.io.pem:

# This certificate is for testing in AWS London region

file: $PWD/certs/ci.pem

version: '3.7'

services:

agent:

image: portainer/agent:latest

environment:

# REQUIRED: Should be equal to the service name prefixed by "tasks." when

# deployed inside an overlay network

AGENT\_CLUSTER\_ADDR: tasks.agent

# AGENT\_PORT: 9001

# LOG\_LEVEL: debug

volumes:

- /var/run/docker.sock:/var/run/docker.sock

- /var/lib/docker/volumes:/var/lib/docker/volumes

networks:

- agent\_network

deploy:

mode: global

placement:

constraints: [node.platform.os == linux]

portainer:

image: portainer/portainer:latest

# command: -H tcp://tasks.agent:9001 --tlsskipverify

command: ["-H", "tcp://tasks.agent:9001", "--tlsskipverify", "--no-auth"]

ports:

- "9000:9000"

volumes:

- portainer\_data:/data

networks:

- agent\_network

deploy:

mode: replicated

replicas: 1

placement:

constraints: [node.role == manager]

networks:

agent\_network:

driver: overlay

attachable: true

volumes:

portainer\_data:

networks:

attachable:

attachable: true

gitlab: {}

proxy: {}

sonarqube: {}

secrets:

cert-xip.io.pem:

file: $PWD/certs/ci.pem

jenkins-pass:

file: $PWD/secrets/jenkins/jenkins-pass.txt

jenkins-user:

file: $PWD/secrets/jenkins/jenkins-user.txt

services:

gitlab:

deploy:

labels:

com.df.distribute: "true"

com.df.httpsOnly.1: "true"

com.df.notify: "true"

com.df.port.1: '80'

com.df.port.2: '22'

com.df.reqMode.2: tcp

com.df.servicePath.1: /gitlab

com.df.srcPort.1: '443'

com.df.srcPort.2: '10022'

environment:

DB\_ADAPTER: postgresql

DB\_HOST: gitlabDB

DB\_NAME: gitlabhq\_production

DB\_PASS: password

DB\_PORT: '5432'

DB\_USER: gitlab

DEBUG: "false"

GITLAB\_BACKUP\_SCHEDULE: daily

GITLAB\_BACKUP\_TIME: 01:00

GITLAB\_EMAIL: notifications@example.com

GITLAB\_EMAIL\_REPLY\_TO: noreply@example.com

GITLAB\_HOST: ${DefaultDNSTarget:-node1}

GITLAB\_HTTPS: "true"

GITLAB\_INCOMING\_EMAIL\_ADDRESS: reply@example.com

GITLAB\_NOTIFY\_ON\_BROKEN\_BUILDS: "true"

GITLAB\_NOTIFY\_PUSHER: "false"

GITLAB\_PORT: '443'

GITLAB\_RELATIVE\_URL\_ROOT: /gitlab

GITLAB\_ROOT\_EMAIL: admin@example.com

GITLAB\_ROOT\_PASSWORD: Password01

GITLAB\_SECRETS\_DB\_KEY\_BASE: long-and-random-alphanumeric-string

GITLAB\_SECRETS\_OTP\_KEY\_BASE: long-and-random-alphanumeric-string

GITLAB\_SECRETS\_SECRET\_KEY\_BASE: long-and-random-alphanumeric-string

GITLAB\_SSH\_PORT: '10022'

REDIS\_HOST: redis

REDIS\_PORT: '6379'

SSL\_SELF\_SIGNED: "true"

hostname: gitlab

image: sameersbn/gitlab:latest

networks:

gitlab: null

proxy: null

volumes:

- gitlab\_data:/home/git/data:rw

gitlabDB:

environment:

DB\_EXTENSION: pg\_trgm

DB\_NAME: gitlabhq\_production

DB\_PASS: password

DB\_USER: gitlab

hostname: gitlabDB

image: sameersbn/postgresql:latest

networks:

gitlab: null

volumes:

- gitlabPostgresql\_data:/var/lib/postgresql:rw

jenkins:

deploy:

labels:

com.df.distribute: "true"

com.df.notify: "true"

com.df.port: '8080'

com.df.servicePath: /jenkins

com.df.srcPort: '443'

placement:

constraints:

- node.role == manager

environment:

JENKINS\_OPTS: '''--prefix=/jenkins'''

hostname: jenkins

image: shazchaudhry/docker-jenkins:latest

networks:

attachable: null

proxy: null

secrets:

- source: jenkins-pass

- source: jenkins-user

user: root

volumes:

- $PWD/maven:/maven:rw

- jenkins\_home:/var/jenkins\_home:rw

- /var/run/docker.sock:/var/run/docker.sock:rw

nexus:

deploy:

labels:

com.df.distribute: "true"

com.df.notify: "true"

com.df.port.1: '8081'

com.df.port.2: '8082'

com.df.port.3: '5000'

com.df.servicePath.1: /nexus

com.df.servicePath.2: /

com.df.servicePath.3: /

com.df.srcPort.1: '443'

com.df.srcPort.2: '443'

com.df.srcPort.3: '5000'

environment:

NEXUS\_CONTEXT: nexus

hostname: nexus

image: sonatype/nexus3:latest

networks:

attachable: null

proxy: null

user: root

volumes:

- nexus\_data:/nexus-data:rw

proxy:

environment:

BIND\_PORTS: '5000'

LISTENER\_ADDRESS: swarm-listener

MODE: swarm

hostname: proxy

image: dockerflow/docker-flow-proxy:latest

networks:

proxy: null

ports:

- published: 80

target: 80

- published: 443

target: 443

- published: 5000

target: 5000

- published: 10022

target: 10022

secrets:

- source: cert-xip.io.pem

redis:

command:

- --loglevel warning

hostname: redis

image: sameersbn/redis:latest

networks:

gitlab: null

volumes:

- redis\_data:/var/lib/redis:rw

sonarDB:

environment:

POSTGRES\_PASSWORD: sonar

POSTGRES\_USER: sonar

hostname: sonarDB

image: postgres:latest

networks:

sonarqube: null

volumes:

- postgresql:/var/lib/postgresql:rw

- postgresql\_data:/var/lib/postgresql/data:rw

sonarqube:

command:

- -Dsonar.web.context=/sonar

deploy:

labels:

com.df.distribute: "true"

com.df.notify: "true"

com.df.port: '9000'

com.df.servicePath: /sonar

com.df.srcPort: '443'

environment:

SONARQUBE\_JDBC\_PASSWORD: sonar

SONARQUBE\_JDBC\_URL: jdbc:postgresql://sonarDB:5432/sonar

SONARQUBE\_JDBC\_USERNAME: sonar

hostname: sonarqube

image: sonarqube:latest

networks:

attachable: null

proxy: null

sonarqube: null

volumes:

- sonarqube\_conf:/opt/sonarqube/conf:rw

- sonarqube\_data:/opt/sonarqube/data:rw

- sonarqube\_extensions:/opt/sonarqube/extensions:rw

- sonarqube\_bundled\_plugins:/opt/sonarqube/lib/bundled-plugins:rw

swarm-listener:

deploy:

placement:

constraints:

- node.role == manager

environment:

DF\_NOTIFY\_CREATE\_SERVICE\_URL: http://proxy:8080/v1/docker-flow-proxy/reconfigure

DF\_NOTIFY\_REMOVE\_SERVICE\_URL: http://proxy:8080/v1/docker-flow-proxy/remove

hostname: swarm-listener

image: dockerflow/docker-flow-swarm-listener:latest

networks:

proxy: null

volumes:

- /var/run/docker.sock:/var/run/docker.sock:rw

version: '3.7'

volumes:

gitlabPostgresql\_data:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

gitlab\_data:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

jenkins\_home:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

nexus\_data:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

postgresql: {}

postgresql\_data: {}

redis\_data: {}

sonarqube\_bundled\_plugins: {}

sonarqube\_conf: {}

sonarqube\_data: {}

sonarqube\_extensions: {}

networks:

attachable:

attachable: true

gitlab: {}

proxy: {}

sonarqube: {}

secrets:

cert-xip.io.pem:

file: $PWD/certs/ci.pem

jenkins-pass:

file: $PWD/secrets/jenkins/jenkins-pass.txt

jenkins-user:

file: $PWD/secrets/jenkins/jenkins-user.txt

services:

gitlab:

deploy:

labels:

com.df.distribute: "true"

com.df.httpsOnly.1: "true"

com.df.notify: "true"

com.df.port.1: '80'

com.df.port.2: '22'

com.df.reqMode.2: tcp

com.df.servicePath.1: /gitlab

com.df.srcPort.1: '443'

com.df.srcPort.2: '10022'

environment:

DB\_ADAPTER: postgresql

DB\_HOST: gitlabDB

DB\_NAME: gitlabhq\_production

DB\_PASS: password

DB\_PORT: '5432'

DB\_USER: gitlab

DEBUG: "false"

GITLAB\_BACKUP\_SCHEDULE: daily

GITLAB\_BACKUP\_TIME: 01:00

GITLAB\_EMAIL: notifications@example.com

GITLAB\_EMAIL\_REPLY\_TO: noreply@example.com

GITLAB\_HOST: ${DefaultDNSTarget:-node1}

GITLAB\_HTTPS: "true"

GITLAB\_INCOMING\_EMAIL\_ADDRESS: reply@example.com

GITLAB\_NOTIFY\_ON\_BROKEN\_BUILDS: "true"

GITLAB\_NOTIFY\_PUSHER: "false"

GITLAB\_PORT: '443'

GITLAB\_RELATIVE\_URL\_ROOT: /gitlab

GITLAB\_ROOT\_EMAIL: admin@example.com

GITLAB\_ROOT\_PASSWORD: Password01

GITLAB\_SECRETS\_DB\_KEY\_BASE: long-and-random-alphanumeric-string

GITLAB\_SECRETS\_OTP\_KEY\_BASE: long-and-random-alphanumeric-string

GITLAB\_SECRETS\_SECRET\_KEY\_BASE: long-and-random-alphanumeric-string

GITLAB\_SSH\_PORT: '10022'

REDIS\_HOST: redis

REDIS\_PORT: '6379'

SSL\_SELF\_SIGNED: "true"

hostname: gitlab

image: sameersbn/gitlab:latest

networks:

gitlab: null

proxy: null

volumes:

- gitlab\_data:/home/git/data:rw

gitlabDB:

environment:

DB\_EXTENSION: pg\_trgm

DB\_NAME: gitlabhq\_production

DB\_PASS: password

DB\_USER: gitlab

hostname: gitlabDB

image: sameersbn/postgresql:latest

networks:

gitlab: null

volumes:

- gitlabPostgresql\_data:/var/lib/postgresql:rw

jenkins:

deploy:

labels:

com.df.distribute: "true"

com.df.notify: "true"

com.df.port: '8080'

com.df.servicePath: /jenkins

com.df.srcPort: '443'

placement:

constraints:

- node.role == manager

environment:

JENKINS\_OPTS: '''--prefix=/jenkins'''

hostname: jenkins

image: shazchaudhry/docker-jenkins:latest

networks:

attachable: null

proxy: null

secrets:

- source: jenkins-pass

- source: jenkins-user

user: root

volumes:

- $PWD/maven:/maven:rw

- jenkins\_home:/var/jenkins\_home:rw

- /var/run/docker.sock:/var/run/docker.sock:rw

nexus:

deploy:

labels:

com.df.distribute: "true"

com.df.notify: "true"

com.df.port.1: '8081'

com.df.port.2: '8082'

com.df.port.3: '5000'

com.df.servicePath.1: /nexus

com.df.servicePath.2: /

com.df.servicePath.3: /

com.df.srcPort.1: '443'

com.df.srcPort.2: '443'

com.df.srcPort.3: '5000'

environment:

NEXUS\_CONTEXT: nexus

hostname: nexus

image: sonatype/nexus3:latest

networks:

attachable: null

proxy: null

user: root

volumes:

- nexus\_data:/nexus-data:rw

proxy:

environment:

BIND\_PORTS: '5000'

LISTENER\_ADDRESS: swarm-listener

MODE: swarm

hostname: proxy

image: dockerflow/docker-flow-proxy:latest

networks:

proxy: null

ports:

- published: 80

target: 80

- published: 443

target: 443

- published: 5000

target: 5000

- published: 10022

target: 10022

secrets:

- source: cert-xip.io.pem

redis:

command:

- --loglevel warning

hostname: redis

image: sameersbn/redis:latest

networks:

gitlab: null

volumes:

- redis\_data:/var/lib/redis:rw

sonarDB:

environment:

POSTGRES\_PASSWORD: sonar

POSTGRES\_USER: sonar

hostname: sonarDB

image: postgres:latest

networks:

sonarqube: null

volumes:

- postgresql:/var/lib/postgresql:rw

- postgresql\_data:/var/lib/postgresql/data:rw

sonarqube:

command:

- -Dsonar.web.context=/sonar

deploy:

labels:

com.df.distribute: "true"

com.df.notify: "true"

com.df.port: '9000'

com.df.servicePath: /sonar

com.df.srcPort: '443'

environment:

SONARQUBE\_JDBC\_PASSWORD: sonar

SONARQUBE\_JDBC\_URL: jdbc:postgresql://sonarDB:5432/sonar

SONARQUBE\_JDBC\_USERNAME: sonar

hostname: sonarqube

image: sonarqube:latest

networks:

attachable: null

proxy: null

sonarqube: null

volumes:

- sonarqube\_conf:/opt/sonarqube/conf:rw

- sonarqube\_data:/opt/sonarqube/data:rw

- sonarqube\_extensions:/opt/sonarqube/extensions:rw

- sonarqube\_bundled\_plugins:/opt/sonarqube/lib/bundled-plugins:rw

swarm-listener:

deploy:

placement:

constraints:

- node.role == manager

environment:

DF\_NOTIFY\_CREATE\_SERVICE\_URL: http://proxy:8080/v1/docker-flow-proxy/reconfigure

DF\_NOTIFY\_REMOVE\_SERVICE\_URL: http://proxy:8080/v1/docker-flow-proxy/remove

hostname: swarm-listener

image: dockerflow/docker-flow-swarm-listener:latest

networks:

proxy: null

volumes:

- /var/run/docker.sock:/var/run/docker.sock:rw

version: '3.7'

volumes:

gitlabPostgresql\_data:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

gitlab\_data:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

jenkins\_home:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

nexus\_data:

driver: cloudstor:aws

driver\_opts:

backing: relocatable

ebstype: gp2

iops: '1000'

size: '25'

postgresql: {}

postgresql\_data: {}

redis\_data: {}

sonarqube\_bundled\_plugins: {}

sonarqube\_conf: {}

sonarqube\_data: {}

sonarqube\_extensions: {}

#!/bin/bash

docker stack rm ci