Setting Up Jenkins Pipeline to Deploy Docker Swarm

project source code

DONE BY : B.UDAY NARASA REDDY

settings-docker.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"
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</localRepos
itory>
    <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/mavenpublic/</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>
```

---

ci-slack.xml

FfmkuvXx4SpXs5p47JPRy0d3RoefZt8YAV/pghAE7gThAWIjtNx7G/X4
dCB2Bwbf7tXtEBr7b/rqvSS3bn1CC+/8A</diagram></mxfile>

Docker-compose.AWS.cloudstor.yml

```
version: "3.7"
 volumes:
gitlabPostgresql_data:
driver: "cloudstor:aws"
driver_opts:
      ebstype: gp2 # https://docs.docker.com/docker-
foraws/persistent-data-volumes/#use-a-unique-volume-
pertask-using-ebs &&
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVo
lumeTypes.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
```

Docker-compose.portainer.yml

```
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     comman
H", "tcp://tasks.agent:9001", "-tlsskipverify", "--no-auth"]     p
-    "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:
```

Docker-compose.visualiser.yml

```
version: "3.7"
 services:
   visualizer:
    image: dockersamples/visualizer     ports:
-    "9999:8080/tcp"     volumes:
-    /var/run/docker.sock:/var/run/docker.sock
deploy:       placement:
        constraints: [node.role == manager]
```

Docker-compose.yml

```
version: "3.7"
```

```
services:
     swarm-
listener:
        image: dockerflow/docker-flow-
swarmlistener:latest         hostname:
swarm-listener         networks:
- proxy         volumes:
            -
"/var/run/docker.sock:/var/run/docker.sock"
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
deploy:
            placement:
                constraints: [node.role == manager]

proxy:
        image: dockerflow/docker-flow-proxy:latest
hostname: proxy         ports:             -
"80:80"
```

```
            - "443:443"
```

```
    -    "5000:5000"
```

```
-    "10022:10022" networks:
```

```
-    proxy          environment:
-    LISTENER_ADDRESS=swarm-listener
-    MODE=swarm              - BIND_PORTS=5000
secrets:
-    cert-xip.io.pem

    # See this blog on how to set up docker registry
(ports 8082 and 5000 are for docker proxy and hosted
repos): https://blog.sonatype.com/using-nexus-3-as-
yourrepository-part-3-docker-images     nexus:
        image: sonatype/nexus3:latest
hostname: nexus         user: root
environment:
-    NEXUS_CONTEXT=nexus         networks:
-    proxy              - attachable
volumes:
-    nexus_data:/nexus-data         deploy:
labels:
-    com.df.notify=true
-    com.df.distribute=true
-    com.df.servicePath.1=/nexus
```

```
-    com.df.port.1=8081
-    com.df.srcPort.1=443
-    com.df.servicePath.2=/
```

```
-    com.df.port.2=8082 # reserved for docker group repo
-    com.df.srcPort.2=443
-    com.df.servicePath.3=/
-    com.df.port.3=5000 # reserved for docker hosted
repo
-    com.df.srcPort.3=5000

sonarDB:
        image: postgres:latest
hostname: sonarDB
environment:
-    POSTGRES_USER=sonar             -
POSTGRES_PASSWORD=sonar         networks:             -
sonarqube         volumes:
-    postgresql:/var/lib/postgresql
-    postgresql_data:/var/lib/postgresql/data

sonarqube:
        image: sonarqube:latest
hostname: sonarqube
environment:
            -
SONARQUBE_JDBC_URL=jdbc:postgresql://sonarDB:5432/sonar
-    SONARQUBE_JDBC_USERNAME=sonar
```

```
-    SONARQUBE_JDBC_PASSWORD=sonar
```

```yaml
networks:
    -    sonarqube
    -    proxy
```

```
-    attachable         volumes:
-    sonarqube_conf:/opt/sonarqube/conf
-    sonarqube_data:/opt/sonarqube/data
-    sonarqube_extensions:/opt/sonarqube/extensions
-    sonarqube_bundled_plugins:/opt/sonarqube/lib/bundledplugins
command: ["-Dsonar.web.context=/sonar"]         deploy:
labels:
-    com.df.notify=true
-    com.df.distribute=true
-    com.df.servicePath=/sonar
-    com.df.port=9000
-    com.df.srcPort=443
     jenkins:
        image: shazchaudhry/docker-jenkins:latest
user: root         hostname: jenkins
environment:
-    JENKINS_OPTS='--prefix=/jenkins'         networks:
-    proxy             - attachable         volumes:
-    /var/run/docker.sock:/var/run/docker.sock
```

```
    -    jenkins_home:/var/jenkins_
home
    -    $PWD/maven:/maven
```

```
        secrets: # See how secrets are used in this jenkins image
https://github.com/shazChaudhry/dockerjenkins/blob/master/config/s
- jenkins-user
- jenkins-pass
#        logging:
#            driver: gelf
#            options:
#                gelf-address: udp://127.0.0.1:12201         deplo
placement:
                constraints: [node.role == manager]             la
- com.df.notify=true
- com.df.distribute=true
- com.df.servicePath=/jenkins
- com.df.port=8080
- com.df.srcPort=443
     redis:
        image: sameersbn/redis:latest         hostname: redis
networks:             - gitlab         volumes:
- redis_data:/var/lib/redis   command: ["--loglevel warning"]
```

```yaml
gitlabDB:
    image: sameersbn/postgresql:latest
```

```
        hostname: gitlabDB
networks:             -
gitlab         volumes:
-    gitlabPostgresql_data:/var/lib/postgresql
environment:
-    DB_USER=gitlab
-    DB_PASS=password
-    DB_NAME=gitlabhq_production
-    DB_EXTENSION=pg_trgm

gitlab:
        image:
sameersbn/gitlab:latest
hostname: gitlab         networks:
- gitlab             - proxy
volumes:
-    gitlab_data:/home/git/data         environment:
-    DEBUG=false
-    DB_ADAPTER=postgresql
-    DB_HOST=gitlabDB
-    DB_PORT=5432
-    DB_USER=gitlab
-    DB_PASS=password
-    DB_NAME=gitlabhq_production
```

```
-    REDIS_HOST=redis
```

```
        -    REDIS_PORT=6379
```

```
-    GITLAB_HTTPS=true
-    SSL_SELF_SIGNED=true

-    GITLAB_HOST=node1
-    GITLAB_PORT=443
-    GITLAB_SSH_PORT=10022
-    GITLAB_RELATIVE_URL_ROOT=/gitlab
-    GITLAB_SECRETS_DB_KEY_BASE=long-and-
randomalphanumeric-string
-    GITLAB_SECRETS_SECRET_KEY_BASE=long-andrandom-
alphanumeric-string
-    GITLAB_SECRETS_OTP_KEY_BASE=long-andrandom-
alphanumeric-string

-    GITLAB_ROOT_PASSWORD=Password01
-    GITLAB_ROOT_EMAIL=admin@example.com
-    GITLAB_NOTIFY_ON_BROKEN_BUILDS=true
-    GITLAB_NOTIFY_PUSHER=false

-    GITLAB_EMAIL=notifications@example.com
-    GITLAB_EMAIL_REPLY_TO=noreply@example.com
            -
GITLAB_INCOMING_EMAIL_ADDRESS=reply@example.com

-    GITLAB_BACKUP_SCHEDULE=daily
```

```
-   GITLAB_BACKUP_TIME=01:00

          # Amazon Web Services (AWS) Remote Backups
```

```
    # - AWS_BACKUPS=true       #  -
AWS_BACKUP_REGION=eu-west-2
```

```
          #   - AWS_BACKUP_BUCKET=
          #   - AWS_BACKUP_ACCESS_KEY_ID=
#   - BACKUP_SECRET_ACCESS_KEY=
deploy:              labels:
- com.df.notify=true
- com.df.distribute=true
- com.df.servicePath.1=/gitlab
- com.df.httpsOnly.1=true
- com.df.port.1=80
- com.df.srcPort.1=443
- com.df.port.2=22
- com.df.srcPort.2=10022
- com.df.reqMode.2=tcp

    # keycloak:
    #     image: jboss/keycloak:latest
#     hostname: keycloak
    #     environment:
    #         - KEYCLOAK_PASSWORD=admin
    #         - KEYCLOAK_USER=admin
    #         - PROXY_ADDRESS_FORWARDING=true
    #     networks:
    #         - proxy
    #     deploy:
```

```
    #         labels:
    #             - com.df.notify=true
    #             - com.df.distribute=true
    #             - com.df.servicePath=/
```

```
#               - com.df.port=8080
```

```
networks:
gitlab:
sonarqube:
    proxy:
attachable:
      attachable: true
 volumes:
gitlabPostgresql_data:
gitlab_data:     jenkins_home:
      # See 'REX-Ray Docker volume plug-ins' documentaion;
volume available across entire docker swarm cluster
      #
https://rexray.readthedocs.io/en/v0.9.0/userguide/docker-
plugins/#elastic-block-service       # driver: rexray/ebs
      # driver_opts:
#   size: 5
nexus_data:
postgresql:
    postgresql_data:
redis_data:
sonarqube_bundled_plugins:
sonarqube_conf:     sonarqube_data:
sonarqube_extensions:
```

```
secrets:
    jenkins-pass:   file:
    $PWD/secrets/jenkins/jenkins-pass.txt
    jenkins-user:
      file: $PWD/secrets/jenkins/jenkins-user.txt
cert-xip.io.pem:
      # This certificate is local testing
file: $PWD/certs/xip.io.pem
```

Docker-stack.yml

```
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-
randomalphanumeric-string
      GITLAB_SECRETS_OTP_KEY_BASE: long-and-random-
```

alphanumeric-string

```
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"
```

```
    GITLAB_SECRETS_SECRET_KEY_BASE: long-and-random
```

```
  com.df.notify: "true"   com.df.port: '8080'
```

```
com.df.servicePath: /jenkins
```

```
      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.srcPort: '443'
```

```
    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     -
```

```yaml
published: 5000
target: 5000
```

```
    - published: 10022 target:
      10022 secrets: - source:
```

```
  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"
```

```
      cert-xip.io.pem
      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/bundledplugins:r
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: {}
```

==========X==========
=