

**Setting Up Jenkins Pipeline to Deploy Docker Swarm project source code**



 ***settings-docker.xml***

<?xml version="1.0" encoding="UTF-8"?>

<settings xmlns=["h](http://maven.apache.org/SETTINGS/1.0.0)t[tp://maven.apache.org/SETTINGS/1.0.0](http://maven.apache.org/SETTINGS/1.0.0)" xmlns:xsi=["h](http://www.w3.org/2001/XMLSchemainstance)t[tp://www.w3.org/2001/XMLSchemainstance](http://www.w3.org/2001/XMLSchemainstance)" xsi:schemaLocation=["http://](http://maven.apache.org/SETTINGS/1.0)m[aven.apache.org/SETTINGS/1.0](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>





direct -->

<!--Enable snapshots for the built in central repo to

<!--all requests to nexus via the mirror -->



<repositories>

<repository>







<id>central</id>



<url>[http://central](http://central/)</url>

<releases><enabled>true</enabled></releases>

<snapshots><enabled>true</enabled></snapshots>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>central</id>

<url>[http://central](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/X 4

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:

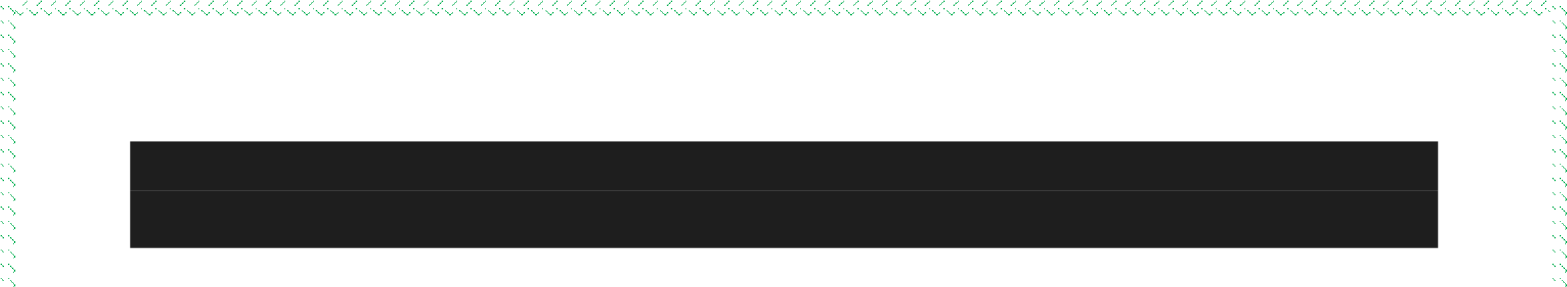
constraints: [node.platform.os == linux] portainer:

image: portainer/portainer:latest

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

- "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

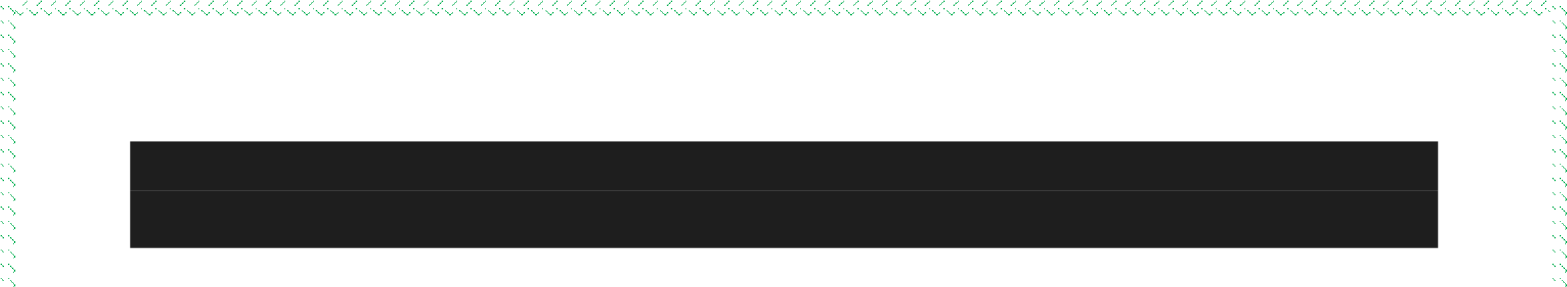
deploy: placement:

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:

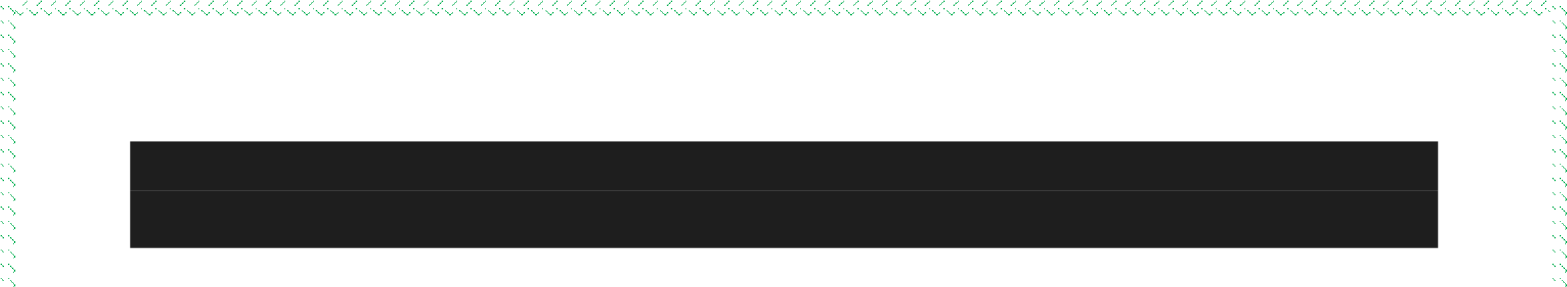
image: dockerflow/docker-flow-proxy:latest

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



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 depl placement:

constraints: [node.role == manager] l

* 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"]



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](mailto:GITLAB_ROOT_EMAIL%3Dadmin@example.com)
* GITLAB\_NOTIFY\_ON\_BROKEN\_BUILDS=true
* GITLAB\_NOTIFY\_PUSHER=false
* [GITLAB\_EMAIL=notifications@example.com](mailto:GITLAB_EMAIL%3Dnotifications@example.com)
* [GITLAB\_EMAIL\_REPLY\_TO=noreply@example.com](mailto:GITLAB_EMAIL_REPLY_TO%3Dnoreply@example.com)

-

[GITLAB\_INCOMING\_EMAIL\_ADDRESS=reply@example.com](mailto:GITLAB_INCOMING_EMAIL_ADDRESS%3Dreply@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

pass:

jenkins-

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](mailto:notifications@example.com) GITLAB\_EMAIL\_REPLY\_TO: [noreply@example.com](mailto:noreply@example.com) GITLAB\_HOST: ${DefaultDNSTarget:-node1} GITLAB\_HTTPS: "true"

GITLAB\_INCOMING\_EMAIL\_ADDRESS: [reply@example.com](mailto: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](mailto: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 -



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===============================

=