



## DEPLOY SUPERSET IN KUBERNETES CLUSTER



---

## Contents

1.	Prerequisites:	4
2.	Installation .....	4
2.1	Clone the deployment files from GitHub.....	4
2.2	Deploy Postgres .....	4
2.3	Deploy Redis.....	5
2.4	Deploy Superset.....	5
2.5	Launch Superset.....	5
3.	Additional Information.....	6
3.1	Integration with Corporate AD .....	6
3.2	Integration with Hive .....	7
3.3	Github login and validate folders & files.....	8
3.4	Modify docker image and push it to repository .....	9
3.5	Setup new superset environment using deployment yaml files .....	10
3.6	Nginx Ingress Setup.....	10



## **Introduction**

This document describes the steps to deploy Superset visualization tool in Kubernetes Cluster.

## 1. Prerequisites:

- Kubernetes Cluster
- NFS Storage
- Nginx Ingress Controller

## 2. Installation

### 2.1 Clone the deployment files from GitHub.

- a) Login to K8S Workstation using PIM. This will login the user as “cpadmin” service account.
- b) Change the user to root using **sudo su**
- c) Create a directory as follows:  
`mkdir superset`
- d) Go inside directory and clone the code from Github Repo as follows:  
`cd superset`  
`git clone http://hyccitlab.ril.com/superset/superset.git`

Note – Check “Additional Information” section to login to Github and get the clone url.

- e) cd superset/production and validate the files (4 folders should exists)

```
root@K8-Pro-Master-1:/home/cpadmin/testsupersetv1/superset/production# ls -lrt
total 16
drwxr-xr-x 2 root root 4096 Sep 26 16:12 kube-redis
drwxr-xr-x 2 root root 4096 Sep 26 16:12 kube-postgres
drwxr-xr-x 2 root root 4096 Sep 26 16:12 superset-yaml
drwxr-xr-x 3 root root 4096 Sep 26 16:12 kube-superset
```

### 2.2 Deploy Postgres

Go inside superset-yaml folder and execute the below commands to deploy postgres as Pod and Service.

- a) Create a new namespace as follows:  
`kubectl create namespace <name of namespace>`
- b) `kubectl apply -f postgres-configmap.yaml <name of new namespace>` - This will create postgres config with credentials.
- c) `kubectl apply -f postgres-storage-nfs.yaml <name of new namespace>` - This will create postgres storage config in nfs.

Note: If there is a change in NFS location, update this yaml file before running the above command. Change the path to the new location.

Example,

hostPath:

path: "/DV\_NFS/postgressdocker"

- d) `kubectl apply -f postgres-deployment.yaml <name of new namespace>` - This will create postgres pod with db.



- e) kubectl apply -f postgres-service.yaml <name of new namespace> - This will create postgres service to be consumed.

## 2.3 Deploy Redis

Go inside superset-yaml folder and execute the below commands to deploy redis as Pod and Service

- a) kubectl apply -f redis-master-deployment.yaml <name of new namespace> - This will create master redis pod with db.
- b) kubectl apply -f redis-master-service.yaml <name of new namespace> - This will create master redis service to be consumed.
- c) kubectl apply -f redis-slave-deployment.yaml <name of new namespace> - This will create slave redis pod with db.
- d) kubectl apply -f redis-slave-service.yaml <name of new namespace> - This will create slave redis service to be consumed.

## 2.4 Deploy Superset

Go inside superset-yaml folder and execute the below commands to deploy superset as Pod and Service.

- a) kubectl apply -f superset-deployment.yaml <name of new namespace> - This will create superset UI pod.
- b) kubectl apply -f superset -service.yaml <name of new namespace> - This will create superset service to be consumed.

## 2.5 Launch Superset

- a) Check the health of the pods created. All pods should be in running state.

kubectl get pods -n <name of new namespace>

```
root@k8s-Pro-Master-1:/home/cpadmin/swappnil/testsupersetv1/superset/production/kube-superset# kubectl get pods -n supersetns090919
NAME                               READY   STATUS    RESTARTS   AGE
redis-master-6fbcb415d67-gtshk   1/1    Running   0          10d
redis-slave-77878d5c67-5bjj2     1/1    Running   0          10d
redis-slave-77878d5c67-z2gg9     1/1    Running   0          10d
superset-deployment-6f7e99dc0ddmgwxs 1/1    Running   0          10d
superset-postgres-0f889057-9m640  1/1    Running   0          10d
root@k8s-Pro-Master-1:/home/cpadmin/swappnil/testsupersetv1/superset/production/kube-superset#
```

- b) Check the health of the deployments created. All services should be in ready state.

kubectl get deployments -n <name of new namespace>

```
root@k8s-Pro-Master-1:/home/cpadmin/swappnil/testsupersetv1/superset/production/kube-superset# kubectl get deployments -n supersetns090919
NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
redis-master                   1/1     1           1           10d
redis-slave                    2/2     2           2           10d
superset-deployment            1/1     1           1           10d
superset-postgres              1/1     1           1           10d
root@k8s-Pro-Master-1:/home/cpadmin/swappnil/testsupersetv1/superset/production/kube-superset#
```

- c) Check the health of the services created. The services should be listed with nodeport assigned.

kubectl get svc -n < name of new namespace >

```
root@k8s-Pro-Master-1:/home/cpadmin/swappnil/testsupersetv1/superset/production/kube-superset# kubectl get svc -n supersetns090919
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
redis-master  ClusterIP  10.101.71.126  <none>          6379/TCP      10d
redis-slave   ClusterIP  10.99.79.222  <none>          6379/TCP      10d
superset     NodePort    10.98.150.197  <none>          8080:30409/TCP  10d
superset-postgres  NodePort    10.100.247.270  <none>          3432:30277/TCP  10d
root@k8s-Pro-Master-1:/home/cpadmin/swappnil/testsupersetv1/superset/production/kube-superset#
```

- d) Run startup script to initialize admin credentials and sample dashboards



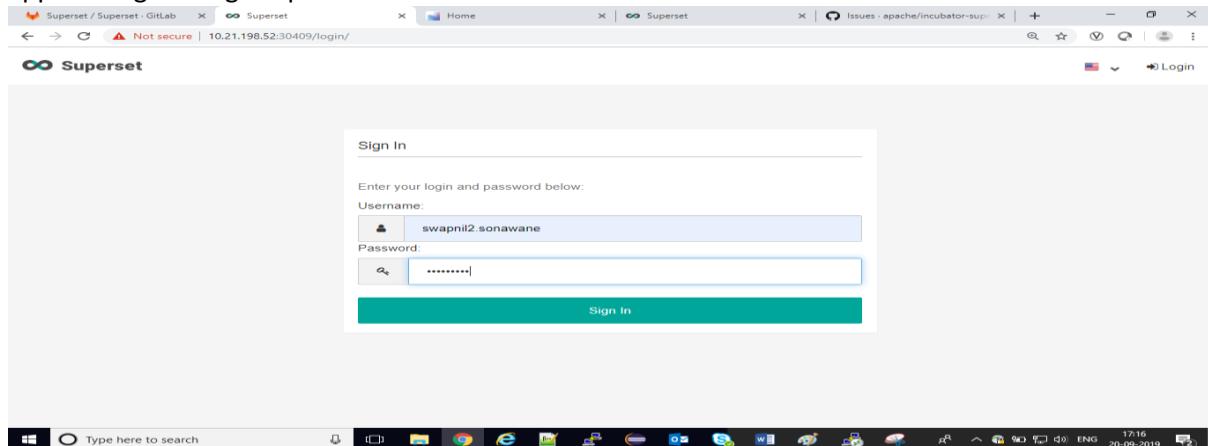
Note – This step is not required if the superset is pointing to the same postgres nfs storage for a running superset instance.

```
Kubectl exec -it -n < name of new namespace > <name of pod> bash  
bash docker-init.sh
```

Example:

```
root@K8-Pro-Master-1:/home/cpadmin/testsupersetv1/superset/production# kubectl exec -it -n superset docker superset-deployment-65b478f98d-4d4kc bash  
superset@superset-deployment-65b478f98d-4d4kc:~$ ls -lrt  
total 44  
-rw-r--r-- 1 root      root    2867 Sep  3 12:10 requirements.txt  
-rw-r--r-- 1 root      root     797 Sep 10 05:40 requirements-extra.txt  
-rwxr-xr-x 1 root      root    1190 Sep 10 05:40 docker-init.sh  
-rwxr--r-- 1 root      root    2525 Sep 10 05:45 krb5.keytab  
-rwxr--r-- 1 root      root    1145 Sep 10 05:45 krb5.conf  
-rwxr--r-- 1 root      root     418 Sep 10 05:45 hive.service.keytab  
-rwxr--r-- 1 root      root    1267 Sep 10 07:04 requirements-dev.txt  
drwxr-xr-x  2 superset  superset 4096 Sep 10 07:23 keytabs  
drwxrwxrwx  1 superset  superset  348 Sep 16 06:03 superset.headless.keytab  
drwxr-xr-x  1 superset  superset 4096 Sep 25 13:27 superset  
superset@superset-deployment-65b478f98d-4d4kc:~$
```

- e) Launch superset using master IP or any other slave IP with node port. Superset login page should appear. Login using Corporate AD credentials.



### 3. Additional Information

This section covers the following additional information:

#### 3.1 Integration with Corporate AD

- a) Login to the superset pod using root user.

```
source ~/.bash_profile
```

```
kubectl ssh -u root -n <name of the new namespace> <name of pod>
```

Note – If there is an error message on “Error: unknown command ‘ssh’ for ‘kubectl’” then pls refer the doc on additional plugin enablement.

<https://github.com/jordanwilson230/kubectl-plugins>



- b) Change config.py as follows:

```
cd /home/superset/superset
```

```
vi config.py
```

Add the following parameters:

```
AUTH_LDAP_BIND_PASSWORD='SuperSecret@123'  
AUTH_LDAP_BIND_USER='CN=HYCDLRILNP.ADMIN,OU=SERVICEACCT,DC=in,DC=ril,DC=com'  
AUTH_LDAP_SEARCH='DC=in,DC=ril,DC=com'  
AUTH_LDAP_SERVER='ldap://adldap.in.ril.com:3268'  
AUTH_LDAP_UID_FIELD='sAMAccountName'  
AUTH_LDAP_USE_TLS=False  
AUTH_LDAP_USERNAME_FORMAT='%s'  
AUTH_TYPE=AUTH_LDAP  
AUTH_USER_REGISTRATION=True  
  
AUTH_USER_REGISTRATION_ROLE='Gamma_Role'
```

### 3.2 Integration with Hive

- a) Initialize Keytab

Go to /etc/security/keytabs

```
Chmod 777 superset.headless.keytab
```

```
Chmod 777 hive.service.keytab
```

```
Chown superset:superset superset.headless.keytab
```

```
kinit -kt /etc/security/keytabs/superset.headless.keytab superset-hcdlprd@HCDLRIL.COM
```

```
klist --- display the Kerberos ticket with expiry and renewal period.
```

- b) Create hive connectivity

Login to superset and Click on Sources -> Databases. (This is available only for users who have admin role in superset)

Create a new database with SQLAlchemy URI as

```
hive://hive@sidcphivmaster03.ril.com:10500/superset1?auth=KERBEROS&kerberos_service_name=hive.
```

Click "Test Connection". If "Seems OK" dialog appears, then all set.

Click Save.



Superset

Edit Database

Detail

Database: Hive

SQLAlchemy URI: hive://hive@sidcpfhivmaster03.ril.com:10500/superset1?auth=KERBEROS&kerberos\_service\_name=hive

Test Connection

hycdv.ril.com says  
Seems OK!

OK

To Test hive connectivity.

Goto SQLLab, select database as "Hive", the schema list should populate.

Superset

Untitled Query

Database: hive Hive

Schema: superset1

```
1 SELECT *
2 FROM
3 student7
```

### 3.3 Github login and validate folders & files

#### 3.3.1 Login to [http://hyccgitlab.ril.com/users/sign\\_in](http://hyccgitlab.ril.com/users/sign_in) using your Corporate AD user name and password.

Sign in · GitLab

Not secure | hyccgitlab.ril.com/users/sign\_in

GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

Active Directory Standard Register

Active Directory Username: swapnil2 sonawane

Password: [redacted]

Remember me

Sign In

Explore Help About GitLab



### 3.3.2 Click on Superset Project.

Projects

Your projects Starred projects Explore projects

All Personal

B BIGDATA-TOOLS / BIGDATA-TOOLS Developer

S Superset / Superset Developer

updated 2 hours ago updated 2 weeks ago

hyccitlab.ril.com/superset/superset

### 3.3.3 Click on Clone and copy the URL from Clone with HTTP.

You won't be able to pull or push project code via SSH until you add an SSH key to your profile

Project ID: 381

No license. All rights reserved 1 Commit 1 Branch 0 Tags 24.6 MB Files

master superset / +

Initial commit Satish.Venkatasubramanian authored 2 weeks ago

Name Last commit Last update

production Initial commit 2 weeks ago

Clone with SSH git@hyccitlab.ril.com:superset/

Clone with HTTP http://hyccitlab.ril.com/superset/superset

Copy URL to clipboard

hyccitlab.ril.com/superset/superset

## 3.4 Modify docker image and push it to repository

If there is an requirement of adding additional python packages/copy any local file to image/..., we need to follow the below steps:

Go inside kube-superset folder.

vi Dockerfile

Make the changes and Save the Dockerfile

Build the docker image using, docker build . -f <path to Dockerfile>



Example: docker build . –f /home/cpadmin/swapnil/testsupersetv1/superset/production/kube-superset/Dockerfile.

After successful build of Docker image push the image in Dockerhub or Nexus. Steps to push image to Docker-Hub.

docker commit < docker container id > [We can get the container id from docker ps] – This command will commit the new docker image.

docker tag < docker image id > <name to tag> - This command will create a new tag/ rename the image.

docker login – Login to docker hub using your userid/pwd.

docker push < docker tag name > - This command will push the new image to docker hub

### 3.5 Setup new superset environment using deployment yaml files

Open the superset-deployment.yaml file

Update the “image:” to the new image uploaded to docker hub/Nexus repository

Apply the changes as follows:

kubectl apply –f superset-deployment.yaml –n <name of the namespace>

Note: The default replica is 10. If we want to change it, open the superset-deployment.yaml file and change the replica value. This needs to be done before the apply command.

Example,

```
spec:  
replicas: 6
```

### 3.6 Nginx Ingress Setup

#### 3.6.1 Check if nginx-ingress-controller pod is running.

kubectl get ing –all-namespaces

```
root@K8-Pro-Master-1:/home/cpadmin/plugins/kubectl-plugins# kubectl get po -n ingress-nginx  
NAME                               READY   STATUS    RESTARTS   AGE  
nginx-ingress-controller-74cd66b57-qghql   1/1     Running   0          6d4h
```

If it is not running then run the below command.

kubectl create namespace ingress-nginx [Note: Create a separate new namespace for nginx-ingress-controller.]

kubectl apply –f superset-nginx-controller.yaml –n ingress-nginx

Note – YAML file available at location where Git repo is cloned.



Ex: /home/cpadmin/testsupersetv1/superset/production/superset-yaml

- 3.6.2 Login to nginx-ingress node using PIM. Change user to root as, sudo su
- 3.6.3 Edit the following file available at location: vim /etc/nginx/sites-enabled/default  
location /  
  
**proxy\_pass <http://192.168.71.22:30534>; → private IP of master with nodeport of ingress**

- 3.6.4 Apply the rules yaml file as,

```
kubectl apply -f superset-ingress.yaml -n <name of the superset new namespace>
```

Note – YAML file available at location where Git repo is cloned.

Ex: /home/cpadmin/testsupersetv1/superset/production/superset-yaml