Upgrading DSS Cloud

Steps to upgrade DSS Cloud (with Kubernetes and User Isolation) to latest version.

For commands with sudo, run as your user (i.e. run as cheryl_abundo) instead of as dataiku.

```
[cheryl_abundo@dss-cloud ~]$
```

For all other commands, run as dataiku.

```
[cheryl_abundo@dss-cloud ~]$ sudo su - dataiku
Last login: Mon Dec 16 05:23:17 UTC 2019 on pts/0
[dataiku@dss-cloud ~]$
```

1. **Check if anyone is logged in to DSS cloud**

If ***no one is connected***, stop DSS (run commands as dataiku)
sudo su - dataiku
./dss_data/bin/dss stop

```
[dataiku@dss-cloud ~]$ ./design/bin/dss stop
Shut down
Waiting for DSS to stop ...
DSS stopped
```

If ***someone is connected***, inform the person/s before stopping DSS,

2. **Clean-up DSS disk**
   *rm -rf /dataiku/dss_data/tmp/**
   *rm -rf /dataiku/dss_data/caches/**
   *rm -rf /dataiku/dss_data/exports/**
   *rm -rf /dataiku/dss_data/diagnosis/**
   *rm -rf /dataiku/dss_data/jobs/**
   *rm -rf /dataiku/dss_data/scenarios/**

3. **Backup data directory**

   a. Check the size of the DSS data directory
      du -hcs /dataiku/dss_data

```
30G     ./design
30G     total
```

   b. Check if there's enough disk space in the VM
      df -h

```
[dataiku@dss-cloud ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs         32G     0   32G   0% /dev
tmpfs            32G     0   32G   0% /dev/shm
tmpfs            32G  625M   31G   2% /run
tmpfs            32G     0   32G   0% /sys/fs/cgroup
/dev/sda1       250G   68G  183G  27% /
tmpfs           6.3G     0  6.3G   0% /run/user/1003
tmpfs           6.3G     0  6.3G   0% /run/user/1004
```

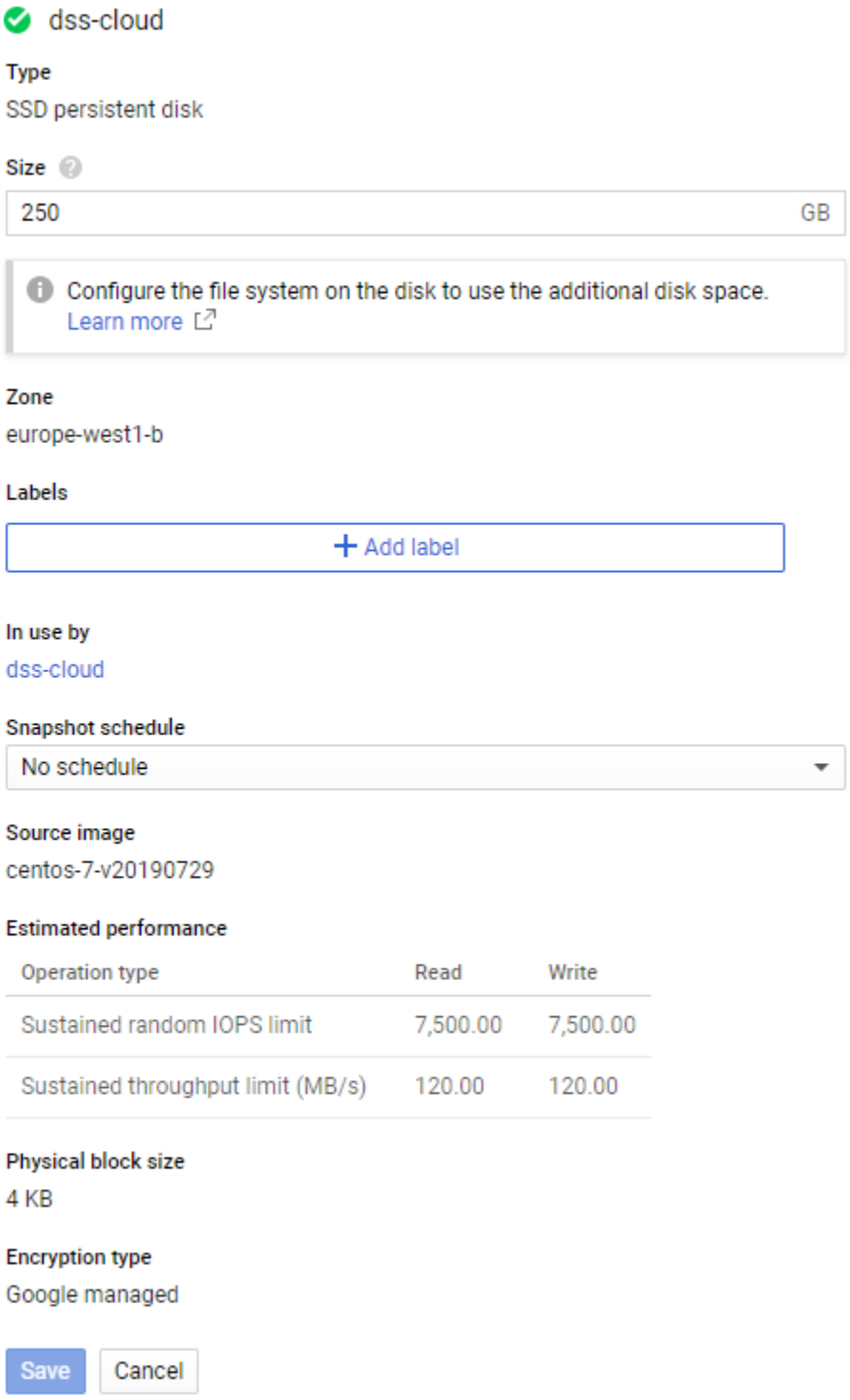
      If there's not enough space, increase disk by following 3c else proceed to 4

   c. Increase disk attached to VM instance

Boot disk

Name	Image	Size (GB)	Device name	Type	Encryption	Mode	When deleting instance
dss-cloud	centos-7-v20190729	250	dss-cloud	SSD persistent disk	Google managed	Boot, read/write	Delete disk

i. Open boot disk and click on *Edit.* Specify higher *Size* and click *Save*.



ii. Open VM SSH and identify the disk with the file system and the partition that you want to resize.
sudo lsblk

iii. Resize the image partition identified above.
sudo growpart /dev/sda 1

iv. Extend the file system on the disk/partition to use the added space.
sudo xfs_growfs /dev/sda1

v. Verify that the file system is resized
df -h /dev/sda1

d. Copy DSS data directory to backup directory (can take awhile, have coffee, work on something else in the meantime)
cp -rv ./dss_data ./dss_data_backup
Check that the backup directory is about the same size (=>) as the original data directory
du -hcs dss_data_backup/
du -hcs dss_data/

4. **Download the latest Dataiku installation file**
wget https://downloads.dataiku.com/public/dss/8.0.1/dataiku-dss-8.0.1.tar.gz
Unpack the file
tar xzf dataiku-dss-8.0.1.tar.gz
After successfully unpacking, delete the tar file
rm dataiku-dss-8.0.1.tar.gz

5. **Perform the upgrade**
dataiku-dss-8.0.1/installer.sh -d dss_data -u

If ***there is a missing dependency***, run the following as your user (ie. cheryl_abundo)
sudo -i "/dataiku/dataiku-dss-6.0.1/scripts/install/install-deps.sh" -without-java -with-conda
Then as dataiku, rerun
dataiku-dss-6.0.1/installer.sh -d design -u

Successful update installation will show

```
************************************************************************
* Installation complete (DSS node type: design)
* Next, start DSS using:
*         '/home/dataiku/design/bin/dss start'
************************************************************************
```

6. **Edit env files**
vim /dataiku/dss_data/env-site.sh

Make sure it includes

```
# This file is sourced last by DSS startup scripts
# You can add local customizations to it
export PATH="/dataiku/anaconda3/condabin:/dataiku/anaconda3/bin:$PATH"
export TEMP="/dataiku/dss_data/tmp"
```

vim /dataiku/dss_data/env-default.sh
Check the following options

```
export DKU_BACKEND_JAVA_OPTS="-Xmx30g -XX:+UseG1GC -Xloggc:/dev/stderr -XX:+PrintGCTimeStamps -Djavax.net.debug=ssl -Djavax.net.ssl.keyStore=/dataiku/certificates/postgresql/client.p12 -Djavax.net.ssl.keyStoreType=pkcs12 -Djavax.net.ssl.keyStorePassword=changeme"
export DKU_FEK_JAVA_OPTS="-Xmx4g -XX:+UseParallelGC -Xloggc:/dev/stderr -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -Djavax.net.debug=ssl -Djavax.net.ssl.keyStore=/dataiku/certificates/postgresql/client.p12 -Djavax.net.ssl.keyStoreType=pkcs12 -Djavax.net.ssl.keyStorePassword=changeme"
export DKU_HPROXY_JAVA_OPTS="-Xmx4g -XX:+UseParallelGC -Xloggc:/dev/stderr -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -Djavax.net.debug=ssl -Djavax.net.ssl.keyStore=/dataiku/certificates/postgresql/client.p12 -Djavax.net.ssl.keyStoreType=pkcs12 -Djavax.net.ssl.keyStorePassword=changeme"
export DKU_JEK_JAVA_OPTS="-Xmx4g -XX:+UseParallelGC -Xloggc:/dev/stderr -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -Djavax.net.debug=ssl -Djavax.net.ssl.keyStore=/dataiku/certificates/postgresql/client.p12 -Djavax.net.ssl.keyStoreType=pkcs12 -Djavax.net.ssl.keyStorePassword=changeme"
```

7. **Update R installation**
./dss_data/bin/dssadmin install-R-integration

```
[InstallKernelSpec] Installed kernelspec ir in /home/dataiku/design/jupyter-run/jupyter/kernels/ir
[+] Creating wrapper script /home/dataiku/design/bin/R
[+] Done
```

8. **Reinstall graphics export**
./dss_data/bin/dssadmin install-graphics-export
If ***there is a missing dependency***, run the following as your user (ie. cheryl_abundo)
sudo -i "/dataiku/dataiku-dss-6.0.1/scripts/install/install-deps.sh" -without-java -without-python -with-chrome
Then as dataiku, rerun
./dss_data/bin/dssadmin install-graphics-export

9. **Reinstall standalone Hadoop and Spark**

a. Download required files
export DSS_VERSION="8.0.1"

wget https://downloads.dataiku.com/public/dss/"$DSS_VERSION"/dataiku-dss-spark-standalone-"$DSS_VERSION"-2.4.5-generic-hadoop3.tar.gz
wget https://downloads.dataiku.com/public/dss/"$DSS_VERSION"/dataiku-dss-hadoop-standalone-libs-generic-hadoop3-"$DSS_VERSION".tar.gz

b. Run Hadoop integration
./dss_data/bin/dssadmin install-hadoop-integration -standaloneArchive /dataiku/dataiku-dss-hadoop-standalone-libs-generic-hadoop3-"$DSS_VERSION".tar.gz

c. Run Spark integration
./dss_data/bin/dssadmin install-spark-integration -standaloneArchive /dataiku/dataiku-dss-spark-standalone-"$DSS_VERSION"-2.4.5-

generic-hadoop3.tar.gz -forK8S

d. Build container images

Download the prebuilt images archive
export DSS_VERSION="8.0.1"
wget http://downloads.dataiku.com/public/dss/"$DSS_VERSION"/container-images/dataiku-dss-ALL-base_dss-"$DSS_VERSION"-r-py3.6.tar.gz

Load the prebuilt images on your local docker repository (on the DSS vm)
gunzip dataiku-dss-ALL-base_dss-"$DSS_VERSION"-r-py3.6.tar.gz
docker image load < dataiku-dss-ALL-base_dss-"$DSS_VERSION"-r-py3.6.tar

Check the new dss-8.0.1 images are loaded
docker images

The following commands will allow your DSS instance to use the DSS images just loaded (double check the name for each container image)
export DSS_CONTAINER_EXEC_IMAGE="dataiku-dss-container-exec-base:dss-"$DSS_VERSION"-r-py3.6"
export DSS_SPARK_IMAGE="dataiku-dss-spark-exec-base:dss-"$DSS_VERSION"r-py3.6"
export DSS_API_NODE_IMAGE="dataiku-dss-apideployer-base:dss-"$DSS_VERSION"-r-py3.6"

./bin/dssadmin build-base-image --type container-exec --mode=use --source-image "$DSS_CONTAINER_EXEC_IMAGE"
./bin/dssadmin build-base-image --type spark --mode=use --source-image "$DSS_SPARK_IMAGE"
./bin/dssadmin build-base-image --type api-deployer --mode=use --source-image "$DSS_API_NODE_IMAGE"

e. *Build code env images*
./dss_data/bin/dssadmin build-code-env-images --all

10. ***Secure the new installation (user isolation** framework)*
Run as your user (ie. cheryl_abundo)
sudo /dataiku/dss_data/bin/dssadmin install-impersonation dataiku

11. ***Start DSS***
./dss_data/bin/dss start

12. ***Configure container execution***

Verify connection by clicking on **TEST**
Click on **PUSH IMAGES**

13. ***Configure Spark***

Click on **PUSH IMAGES**

Related articles

- ALB Data Enrichment - Add new data transformation in the Presentation Layer
- ALB Data Ingestion - Add new data to the data lake
- ALB Operational - deploy to Production
- Writing a simple BW query
- Retrieving columns and variables names for a BW query