**MapR Bigdata Cluster Configuration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Node | CLDB | ZOOKEEPER | NFS | WEBSERVER/APIserver | Fileserver |
| Master01 |  |  |  |  |  |
| S1 |  |  |  |  |  |
| S2 |  |  |  |  |  |
| S3 |  |  |  |  |  |
| S4 |  |  |  |  |  |
| S5 |  |  |  |  |  |

cat /etc/hosts

echo " 10.142.0.8 master01.c.voltaic-racer-208109.internal master01" >> /etc/hosts echo " 10.142.0.9 s1.c.voltaic-racer-208109.internal s1" >> /etc/hosts

echo " 10.142.0.10 s2.c.voltaic-racer-208109.internal s2" >> /etc/hosts

echo " 10.142.0.11 s3.c.voltaic-racer-208109.internal s3" >> /etc/hosts

echo " 10.142.0.12 s4.c.voltaic-racer-208109.internal s4" >> /etc/hosts

echo " 10.142.0.13 s5.c.voltaic-racer-208109.internal s5" >> /etc/hosts

echo " 10.142.0.14 nhost6.c.voltaic-racer-208109.internal nhost6" >> /etc/hosts

echo " 10.142.0.3 s6.c.voltaic-racer-208109.internal s6" >> /etc/hosts

yum update -y

yum install -y curl device-mapper iputils libsysfs lvm2 nc nfs-utils ntp nss openssl python-devel sdparm syslinux sysstat wget yum-utils

wget --no-cookies --no-check-certificate --header "Cookie: oraclelicense=accept-securebackup-cookie" https://download.oracle.com/otn-pub/java/jdk/11.0.1+13/90cf5d8f270a4347a95050320eef3fb7/jdk-11.0.1\_linux-x64\_bin.rpm

curl -v -j -k -L -H "Cookie: oraclelicense=accept-securebackup-cookie" http://download.oracle.com/otn-pub/java/jdk/8u131-b11/d54c1d3a095b4ff2b6607d096fa80163/jdk-8u131-linux-x64.rpm > jdk-8u112-linux-x64.rpm

rpm -ivh jdk-8u181-linux-x64.rpm

env\_override.sh

export JAVA\_HOME=/usr/java/jdk1.8.0\_171-amd64

export MAPR\_HOME=/opt/mapr

export MAPR\_USER=mapr

groupadd -g 5000 mapr

useradd -g 5000 -u 5000 mapr

useradd -m -u 5000 -g 5000 -G $(stat -c '%G' /etc/shadow) mapr

passwd mapr

gpasswd -a mapr mapr

vi /etc/sudoers

mapr ALL=(ALL) ALL

vi /etc/ssh/sshd\_config vi /etc/ssh/ssh\_config

systemctl restart sshd

ssh-copy-id -i /root/.ssh/id\_rsa.pub 10.142.0.3:/root/.ssh/authorized\_keys

vi /etc/yum.repos.d/maprtech.repo

[maprtech] name=MapR Technologies baseurl=http://package.mapr.com/releases/v6.0.1/redhat/ enabled=1 gpgcheck=1

[maprecosystem] name=MapR Technologies

baseurl=http://package.mapr.com/releases/MEP/MEP-5.0.0/redhat enabled=1 gpgcheck=1

protect=1

rpm --import http://package.mapr.com/releases/pub/maprgpg.key

Master01

yum install -y mapr-fileserver mapr-nfs mapr-cldb mapr-resourcemanager mapr-webserver mapr-apiserver mapr-collectd mapr-nodemanager

s1

yum install -y mapr-fileserver mapr-nfs mapr-zookeeper mapr-nodemanager mapr-resourcemanager mapr-collectd

S2

yum install -y mapr-fileserver mapr-nfs mapr-zookeeper mapr-nodemanager mapr-collectd

s3

yum install -y mapr-fileserver mapr-nfs mapr-zookeeper mapr-nodemanager mapr-collectd

s4 and s5

yum install -y mapr-fileserver mapr-nfs mapr-nodemanager mapr-collectd mapr-grafana

yum install -y mapr-fileserver mapr-nfs mapr-nodemanager mapr-collectd mapr-opentsdb

From Master01

Run in master01:

/opt/mapr/server/configure.sh -secure -genkeys -Z s1.c.voltaic-racer-208109.internal:5181,s2.c.voltaic-racer-

208109.internal:5181,s3.c.voltaic-racer-208109.internal:5181 -C master01.c.voltaic-racer-208109.internal:7222 -RM master01.c.voltaicracer-208109.internal:8090,s1.c.voltaic-racer-208109.internal:8090 -N maprcluster -u mapr -g mapr

cd /opt/mapr/conf

scp -r cldb.key ssl\_truststore ssl\_keystore maprserverticket mapr@10.142.0.(9-14):/tmp/

All S(1-5) nodes

mv ssl\_truststore ssl\_keystore maprserverticket cldb.key /opt/mapr/conf/

sudo chmod 600 ssl\_truststore ssl\_keystore maprserverticket cldb.key

(for Grafana 644 permission For a **secured cluster**, copy the /opt/mapr/conf/ssl\_truststore.pem file from the CLDB master node to /opt/mapr/conf on all nodes(or, at a minimum, to the Grafana nodes) )

Run in all S(1-5) Nodes

/opt/mapr/server/configure.sh -secure -Z s1.c.voltaic-racer-208109.internal:5181,s2.c.voltaic-racer-208109.internal:5181, s3.c.voltaic-racer-208109.internal:5181 -C master01.c.voltaic-racer-208109.internal:7222 -N maprcluster -u mapr -g mapr

vi disks /dev/sdb

/dev/sdc

/dev/sdd

/opt/mapr/server/disksetup -FW 1 disks

First Start zookeeper Service

systemctl start mapr-zookeeper

systemctl status mapr-zookeeper

2nd Warden Sevice need to be start

systemctl start mapr-warden

systemctl status mapr-warden

systemctl restart mapr-zookeeper

systemctl restart mapr-warden

**mapr-posix-client-basic**

yum update -y

yum install -y curl device-mapper iputils libsysfs lvm2 nc nfs-utils ntp nss openssl python-devel sdparm syslinux sysstat wget yum-utils

wget --no-cookies --no-check-certificate --header "Cookie: oraclelicense=accept-securebackup-cookie" http://download.oracle.com/otn-pub/java/jdk/8u181-b13/96a7b8442fe84 8ef90c96a2fad6ed6d1/jdk-8u181-linux-x64.rpm

rpm -ivh jdk-8u181-linux-x64.rpm

vi /etc/yum.repos.d/maprtech.repo

[maprtech]

sssname=MapR Technologies

baseurl=http://package.mapr.com/releases/v6.0.1/redhat/ enabled=1 gpgcheck=1

[maprecosystem] name=MapR Technologies

baseurl=http://package.mapr.com/releases/MEP/MEP-5.0.0/redhat enabled=1 gpgcheck=1 protect=1

rpm --import http://package.mapr.com/releases/pub/maprgpg.key

yum install mapr-posix-client-basic

vi /etc/hosts

echo " 10.142.0.8 master01.c.voltaic-racer-208109.internal master01" >> /etc/hosts echo " 10.142.0.9 s1.c.voltaic-racer-208109.internal s1" >> /etc/hosts echo " 10.142.0.10 s2.c.voltaic-racer-208109.internal s2" >> /etc/hosts echo " 10.142.0.11 s3.c.voltaic-racer-208109.internal s3" >> /etc/hosts echo " 10.142.0.12 s4.c.voltaic-racer-208109.internal s4" >> /etc/hosts echo " 10.142.0.13 s5.c.voltaic-racer-208109.internal s5" >> /etc/hosts

groupadd -g 5000 mapr useradd -g 5000 -u 5000 mapr

useradd -m -u 5000 -g 5000 -G $(stat -c '%G' /etc/shadow) mapr passwd mapr

vi /etc/ssh/sshd\_config vi /etc/ssh/ssh\_config

vi /opt/mapr/conf/fuse.conf

#set path to mapr login ticket

fuse.ticketfile.location=/opt/mapr/conf/longlived\_ticket

#Set path to the mount point fuse.mount.point=/mapr

# Create Directory

mkdir /mapr

# Any cluster node run the command

maprlogin generateticket -type service -out /tmp/longlived\_ticket -user mapr

scp -p longlived\_ticket mapr@10.142.0.14:/tmp/ (copy to posix client node)

# From posix client node

cd /tmp/ ls

mv longlived\_ticket /opt/mapr/conf

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/opt/mapr/lib

/opt/mapr/server/configure.sh -secure -Z s1.c.voltaic-racer-208109.internal:5181,s2.c.voltaic-racer-208109.internal:5181,s3.c.voltaic-racer-208109.internal:5181 -C master01.c.voltaic-racer-208109.internal:7222 -N maprcluster -u mapr -g mapr -c

**Run service**

service mapr-posix-client-basic status

service mapr-posix-client-basic start

service mapr-posix-client-basic status

**Check the Directory and verify**  cd /mapr

Latest change in Posix: fuse.conf

----

fuse.num.libs = 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | fuse.cluster.conf.location |  |  | |  | The path to the configuration file to use. |
|  | /opt/mapr/conf/mapr-clusters.conf |
|  |  |

**Hive Configuration**

Install

yum install mapr-hive mapr-hiveserver2 mapr-hivemetastore mapr-hivewebhcat

# Database

yum -y install mariadb-server

yum install mysql-connector-java

## Edit the configuration file

cd /opt/mapr/hive/hive-2.1/conf

vi hive-site.xml

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://n3.us-east1-b.c.main-form-217005.internal:3306/metastore</value>

<description>JDBC connect string for a JDBC metastore</description>

</property>

<property>

<name>javax.jdo.option.ConnectionDriverName</name>

<value>com.mysql.jdbc.Driver</value>

<description>Driver class name for a JDBC metastore</description>

</property>

<property>

<name>javax.jdo.option.ConnectionUserName</name>

<value>hive</value>

<description>username to use against metastore database</description>

</property>

<property>

<name>javax.jdo.option.ConnectionPassword</name>

<value>hivepwd</value>

<description>password to use against metastore database</description>

</property>

<property>

<name>hive.metastore.uris</name>

<value>thrift://n3.us-east1-b.c.main-form-217005.internal:9083</value> </property>

Add the below parameter

<property>

  <name>hive.exec.scratchdir</name>

  <value>/tmp</value>

  <description>Scratch space for Hive jobs</description>

</property>

<property>

  <name>hive.exec.local.scratchdir</name>

  <value>/tmp</value>

  <description>Local scratch space for Hive jobs</description>

</property>

**Db service Start**

Systemctl start mariadb

## Create Database,user and give Grant access

#mysql

mysql> create database metastore; mysql> use metastore;

mysql> CREATE USER 'hive'@'localhost' IDENTIFIED BY 'hivepwd';

...

mysql> REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'hive'@'localhost';

mysql> GRANT ALL PRIVILEGES ON metastore.\* TO 'hive'@'localhost'; mysql> FLUSH PRIVILEGES;

mysql> quit;

and

mysql> CREATE USER 'hive'@'n2' IDENTIFIED BY 'hivepwd';

...

mysql> REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'hive'@'n2';

mysql> GRANT ALL PRIVILEGES ON metastore.\* TO 'hive'@'n2'; mysql> FLUSH PRIVILEGES;

mysql> quit;

and

mysql> CREATE USER 'hive'@'n3.us-east1-b.c.main-form-217005.internal' IDENTIFIED BY 'hivepwd';

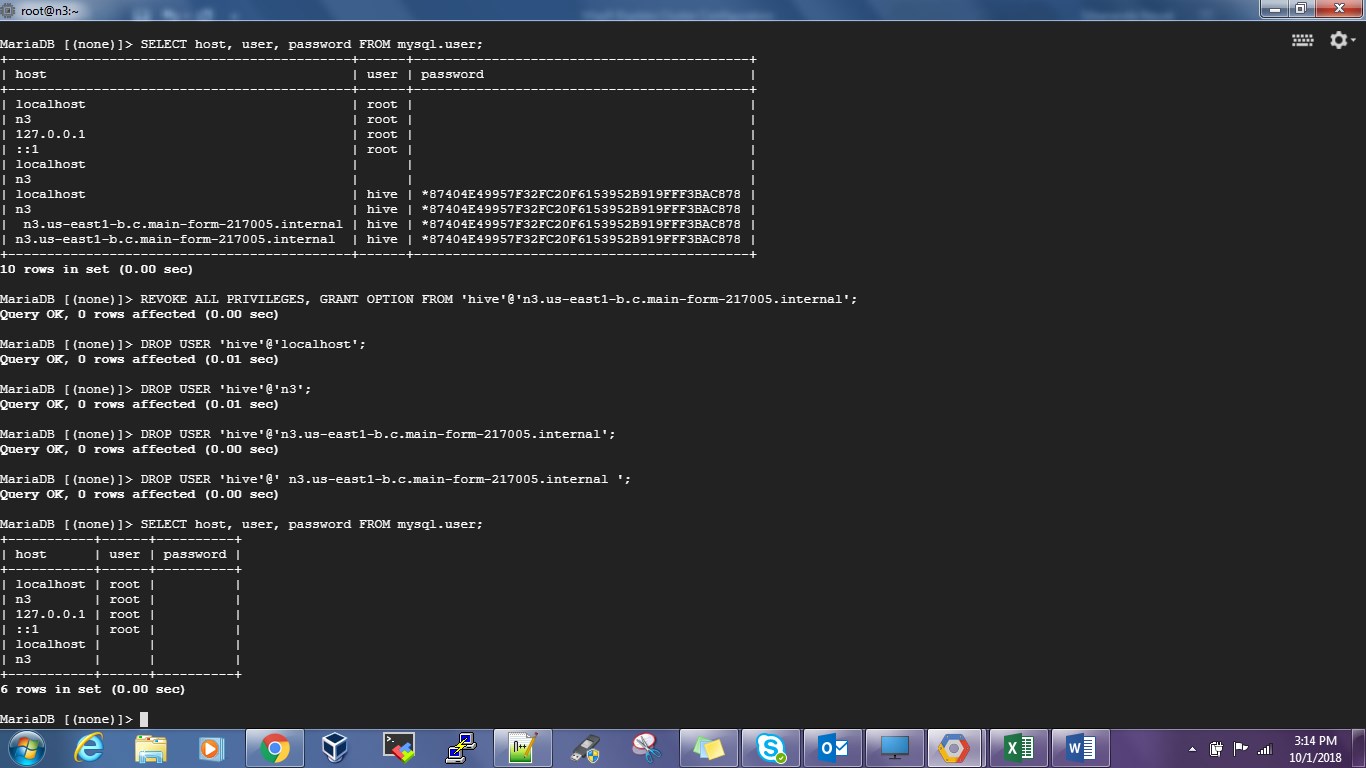
mysql> REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'hive'@'n3.us-east1-b.c.main-form-217005.internal'; mysql> GRANT ALL PRIVILEGES ON metastore.\* TO 'hive'@'n3.us-east1-b.c.main-form-217005.internal'; mysql> FLUSH PRIVILEGES; mysql> quit;

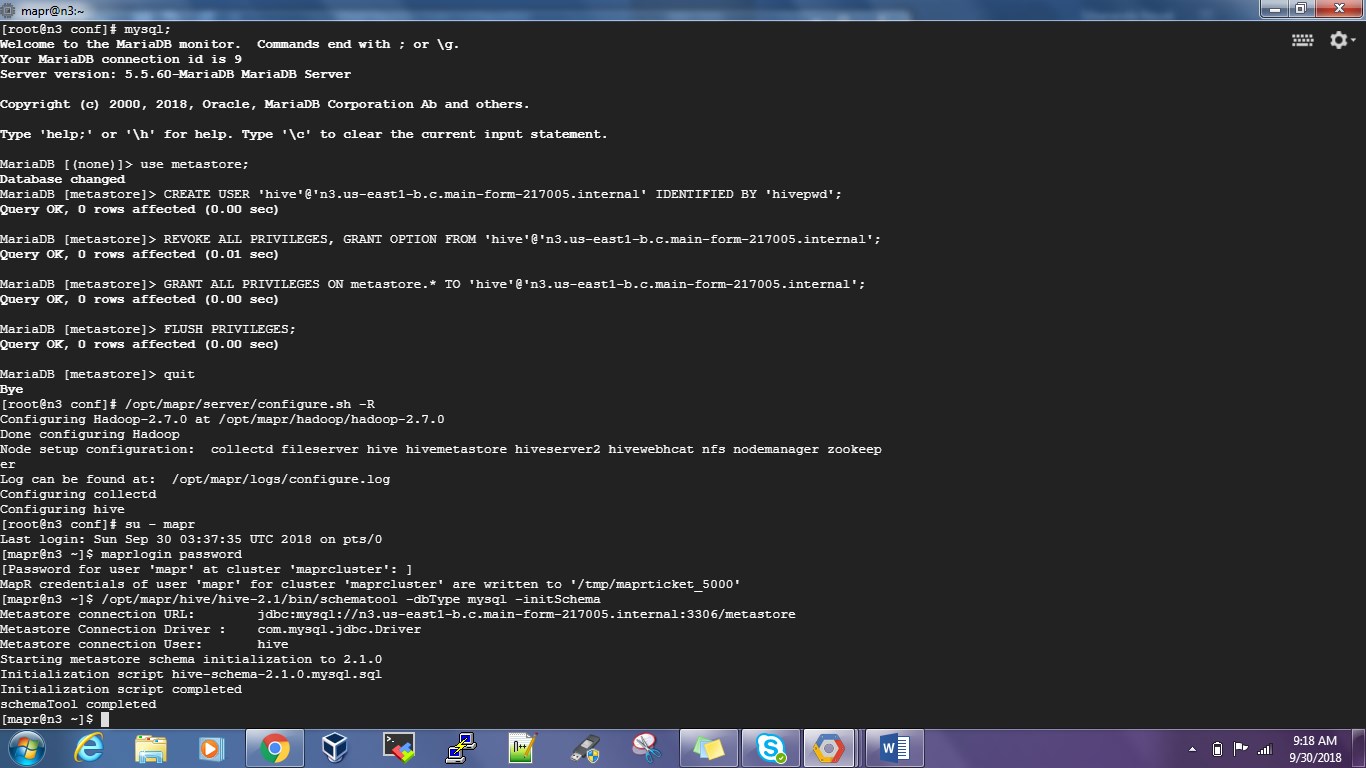
SELECT host, user, password FROM mysql.user;

Drop databasemetastore;

Check for Login

#mysql -u hive -h s6.c.voltaic-racer-208109.internal -p





Configure

/opt/mapr/server/configure.sh -R

systemctl restart mapr-zookeeper systemctl status mapr-zookeeper

systemctl restart mapr-warden

systemctl status mapr-warden

/opt/mapr/hive/hive-2.1/bin/schematool -dbType mysql -initSchema All HIVE service need to start

systemctl start mapr-hiveserver2

Systemctl start mapr-hivemetastore

Systemctl start mapr-hivewebhcat

# hive

**Disk adding and data encryption**

cryptsetup --batch-mode --use-random luksFormat /dev/sdg /etc/crypto/lukskey.bin

cryptsetup luksOpen /dev/sdg luks-sdg < /etc/crypto/lukskey.bin

--

/opt/mapr/server/mrconfig sp list -v

/opt/mapr/server/disksetup -F -W 4 /home/mapr/disklist

Or

cryptsetup --batch-mode --use-random luksFormat /dev/sdq /etc/crypto/lukskey.bin

cryptsetup luksOpen /dev/sdq luks-sdq < /etc/crypto/lukskey.bin

check if added luks --> fdisk -l |grep /dev/sdq

<property>

<name>hive.metastore.sasl.enabled</name>

<value>false</value>

</property>

<https://mapr.com/docs/60/Spark/IntegrateSparkSQL_Hive.html>

Hive Upgrade

Upgrade hive -2.1 to hive -2.3

[root@n2 tmp]# cat /etc/yum.repos.d/maprtech.repo

[maprtech] name=MapR Technologies

baseurl=http://package.mapr.com/releases/v6.1.0/redhat/ enabled=1 gpgcheck=1

[maprecosystem] name=MapR Technologies

baseurl=http://package.mapr.com/releases/MEP/MEP-6.0.0/redhat enabled=1 gpgcheck=1 protect=1

-----

yum update mapr-hive\* -y

-----

cp hive-site.xml ../hive-2.3/conf

cp hive-site.xml /opt/mapr/hive/hive2.3/conf/ pwd

cp hive-site.xml /opt/mapr/hive/hive-2.3/conf pwd

ls -lrt

cd .. cd hive-2.3/ ls -lrt

cd /opt/mapr/hive/hive-2.1/lib cd /opt/mapr/hive/hive-2.1/

ls -lrt cd ..

ls -lrt cd hive-2.1.201804020853

ls -lrt -----

cd /usr/share/java

ls -lrt

cp mysql-connector-java.jar /opt/mapr/hive/hive-2.3/lib

#/opt/mapr/server/configure.sh -R

---- su mapr mysql

/opt/mapr/hive/hive-2.3/scripts/metastore/upgrade/mysql/

[root@n3 mysql]# mysql

MariaDB [(none)]> use metastore

Reading table information for completion of table and column names You can turn off this feature to get a quicker startup with -A

MariaDB [metastore]> select \* from VERSION;

+--------+----------------+----------------------------+

| VER\_ID | SCHEMA\_VERSION | VERSION\_COMMENT |

+--------+----------------+----------------------------+

| 1 | 2.3.0 | Hive release version 2.3.0 |

+--------+----------------+----------------------------+

1 row in set (0.00 sec)

MariaDB [metastore]> source /opt/mapr/hive/hive-2.3/scripts/metastore/upgrade/mysql/upgrade-2.1.0-to-2.2.0.mysql.sql

MariaDB [metastore]> source /opt/mapr/hive/hive-2.3/scripts/metastore/upgrade/mysql/upgrade-2.2.0-to-2.3.0.mysql.sql

@@@@@@@@@@@@@@@ Elastic Search Indeces Issue@@@@@@@@@@@@@@@@@@@@@

<https://mapr.com/support/s/article/Log-retention-doesn-t-work?language=en_US>

[root@esekilxgp10 enaysib]# cd /opt/mapr/elasticsearch/elasticsearch-5.4.1/usr/share/curator/ [root@esekilxgp10 curator]#

[root@esekilxgp10 curator]# find . -type d -exec chmod +x {} \;

[root@esekilxgp10 curator]# ls -ltr total 2128

-rw-r--r-- 1 mapr maprg 1843136 Oct 28 2017 libpython2.7.so.1.0

-rw-r--r-- 1 mapr maprg 10488 Oct 28 2017 es\_repo\_mgr

-rw-r--r-- 1 mapr maprg 10488 Oct 28 2017 curator\_cli

-rw-r--r-- 1 mapr maprg 10488 Oct 28 2017 curator -rw-r--r-- 1 mapr maprg 296399 Oct 28 2017 cacert.pem drwxr-xr-x 3 mapr maprg 43 Jan 23 2018 lib drwxr-xr-x 3 mapr maprg 43 Jan 23 2018 lib64 [root@esekilxgp10 curator]# chmod +x curator curator\_cli

[root@esekilxgp10 curator]# cd /opt/mapr/elasticsearch/elasticsearch-5.4.1/bin

[root@esekilxgp10 bin]# ln -s /opt/mapr/elasticsearch/elasticsearch-5.4.1/usr/share/curator/curator curator

[root@esekilxgp10 bin]# vi /opt/mapr/elasticsearch/elasticsearch-5.4.1/etc/elasticsearch/curator.yml [root@esekilxgp10 bin]# cat /opt/mapr/elasticsearch/elasticsearch-5.4.1/etc/elasticsearch/curator.yml ---

# Remember, leave a key empty if there is no value. None will be a string,

# not a Python "NoneType" client: hosts:

- esekilxgp10.rnd.ki.sw.ericsson.se port: 9200 url\_prefix: use\_ssl: True certificate: client\_cert: client\_key: aws\_key:

/

aws\_secret\_key: aws\_region:

ssl\_no\_validate: True http\_auth: admin:admin timeout: 30 master\_only: False

logging:

loglevel: INFO logfile:

logformat: default

blacklist: ['elasticsearch', 'urllib3'] [root@esekilxgp10 bin]# crontab -u mapr -l

SHELL=/bin/bash

15 3 \* \* \* /opt/mapr/elasticsearch/elasticsearch-5.4.1/bin/curator --config /opt/mapr/elasticsearch/elasticsearch-5.4.1

/opt/mapr/elasticsearch/elasticsearch-5.4.1/etc/elasticsearch/curator\_actions/delete\_indices.yml >> /opt/mapr/elasticsearch/e

[root@esekilxgp10 bin]# su mapr

[mapr@esekilxgp10 bin]$ maprlogin password

[Password for user 'mapr' at cluster 'rdidev1': ]

MapR credentials of user 'mapr' for cluster 'rdidev1' are written to '/tmp/maprticket\_123400016'

[mapr@esekilxgp10 bin]$ /opt/mapr/elasticsearch/elasticsearch-5.4.1/bin/curator --config /opt/mapr/elasticsearch/elasticsearc

/opt/mapr/elasticsearch/elasticsearch-5.4.1/etc/elasticsearch/curator\_actions/delete\_indices.yml >> /opt/mapr/elasticsearch/

[mapr@esekilxgp10 bin]$ exit

[root@esekilxgp10 bin]# df -H /opt/

Filesystem Size Used Avail Use% Mounted on

/dev/mapper/rootvg-rootvol 500G 193G 307G 39% /

[root@esekilxgp10 bin]#

curl -k -u admin:admin -XDELETE https://esekilx5646.rnd.ki.sw.ericsson.se:9200/mapr\_monitoring-2019.03.\*

----------R and sparkR and Spark and hive ---------------

[root@esekilxgp05 ~]# yum install R

[root@esekilxgp05 ~]# which R

Set $HOME in /usr/bin and its show in env

[root@esekilxgp05 ~]# cat .bash\_profile

# .bash\_profile

e

# Get the aliases and functions if [ -f ~/.bashrc ]; then . ~/.bashrc fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH

[root@esekilxgp05 ~]# env

PATH=/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/usr/java/jdk1.8.0\_161//bin:/usr/java/

For sparkR and Spark and hive--------------------

[root@esekilxgp05 ~]# yum install -y mapr-spark mapr-hive

[root@esekilxgp05 ~]# cd /opt/mapr/spark/spark-2.1.0/conf/

[root@esekilxgp05 ~]#ls -ltar

-rwxr-xr-x 1 mapr maprg 7191 Oct 11 07:51 spark-env.sh

-rw-r--r-- 1 mapr maprg 3727 Oct 11 07:51 hive-site.xml

-rw-r--r-- 1 mapr maprg 1749 Oct 11 07:52 spark-defaults.conf

[root@esekilxgp05 ~]# vi spark-defaults.conf

Add below files:

spark.yarn.dist.files = /opt/mapr/spark/spark-2.3.1/conf/hive-site.xml

spark.sql.hive.metastore.version = 1.2.0

[root@esekilxgp05 conf]# cat hive-site.xml

<?xml version="1.0"?>

<!--

Licensed to the Apache Software Foundation (ASF) under one or more contributor license agreements. See the NOTICE file distributed with this work for additional information regarding copyright ownership. The ASF licenses this file to you under the Apache License, Version 2.0 (the

"License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

-->

<configuration>

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://esekilxgp06.rnd.ki.sw.ericsson.se:3306/hive?createDatabaseIfNotExist=true</value> <description>JDBC connect string for a JDBC metastore</description>

</property>

<property>

<name>javax.jdo.option.ConnectionDriverName</name>

<value>com.mysql.jdbc.Driver</value>

<description>Driver class name for a JDBC metastore</description> </property>

<property>

<name>javax.jdo.option.ConnectionUserName</name>

<value>hive</value>

<description>username to use against metastore database</description> </property>

<property>

<name>javax.jdo.option.ConnectionPassword</name>

<value>CyT4cPPfwDs</value>

<description>password to use against metastore database</description>

</property>

<property>

<name>hive.metastore.schema.verification</name>

<value>false</value>

</property>

<property>

<name>hive.metastore.uris</name>

<value>thrift://esekilx5636.rnd.ki.sw.ericsson.se:9083,thrift://esekilx5637.rnd.ki.sw.ericsson.se:9083</value>

</property>

<!-- For hive server2 -->

<property>

<name>hive.server2.enable.doAs</name>

<value>true</value>

</property>

<!-- For hive server2 and meta store -->

<property>

<name>hive.metastore.execute.setugi</name>

<value>true</value>

</property>

<property>

<name>hive.metastore.warehouse.dir</name>

<value>/project/rdi/warehouse/hive</value>

<description>location of default database for the warehouse</description> </property>

<property>

<name>hive.metastore.try.direct.sql</name>

<value>true</value>

<description>

Whether the Hive metastore should try to use direct SQL queries instead of the

DataNucleus for certain read paths. This can improve metastore performance when fetching many partitions or column statistics by orders of magnitude; however, it is not guaranteed to work on all RDBMS-es and all versions. In case of SQL failures, the metastore will fall back to the DataNucleus, so it's safe even if SQL doesn't work for all queries on your datastore. If all SQL queries fail (for example, your metastore is backed by MongoDB), you might want to disable this to save the try-and-fall-back cost.

</description>

</property>

<property>

<name>hive.metastore.client.socket.timeout</name>

<value>1800s</value>

<description>

Expects a time value with unit (d/day, h/hour, m/min, s/sec, ms/msec, us/usec, ns/nsec), which is sec if not specified.

MetaStore Client socket timeout in seconds

</description>

</property>

<property>

<name>hive.metastore.sasl.enabled </name>

<value>false</value>

</property>

</configuration>

[root@esekilxgp05 conf]# cat spark-env.sh

#!/usr/bin/env bash

#

# Licensed to the Apache Software Foundation (ASF) under one or more

# contributor license agreements. See the NOTICE file distributed with # this work for additional information regarding copyright ownership.

# The ASF licenses this file to You under the Apache License, Version 2.0

# (the "License"); you may not use this file except in compliance with

# the License. You may obtain a copy of the License at #

# http://www.apache.org/licenses/LICENSE-2.0

#

# Unless required by applicable law or agreed to in writing, software

# distributed under the License is distributed on an "AS IS" BASIS,

# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. # See the License for the specific language governing permissions and # limitations under the License.

#

# This file is sourced when running various Spark programs.

# Copy it as spark-env.sh and edit that to configure Spark for your site.

# Options read when launching programs locally with

# ./bin/run-example or ./bin/spark-submit

# - HADOOP\_CONF\_DIR, to point Spark towards Hadoop configuration files

# - SPARK\_LOCAL\_IP, to set the IP address Spark binds to on this node

# - SPARK\_PUBLIC\_DNS, to set the public dns name of the driver program # - SPARK\_CLASSPATH, default classpath entries to append

# Options read by executors and drivers running inside the cluster

# - SPARK\_LOCAL\_IP, to set the IP address Spark binds to on this node

# - SPARK\_PUBLIC\_DNS, to set the public DNS name of the driver program

# - SPARK\_CLASSPATH, default classpath entries to append

# - SPARK\_LOCAL\_DIRS, storage directories to use on this node for shuffle and RDD data # - MESOS\_NATIVE\_JAVA\_LIBRARY, to point to your libmesos.so if you use Mesos

# Options read in YARN client mode

# - HADOOP\_CONF\_DIR, to point Spark towards Hadoop configuration files

# - SPARK\_EXECUTOR\_INSTANCES, Number of executors to start (Default: 2) # - SPARK\_EXECUTOR\_CORES, Number of cores for the executors (Default: 1).

# - SPARK\_EXECUTOR\_MEMORY, Memory per Executor (e.g. 1000M, 2G) (Default: 1G)

# - SPARK\_DRIVER\_MEMORY, Memory for Driver (e.g. 1000M, 2G) (Default: 1G)

# Options for the daemons used in the standalone deploy mode

# - SPARK\_MASTER\_HOST, to bind the master to a different IP address or hostname

# - SPARK\_MASTER\_PORT / SPARK\_MASTER\_WEBUI\_PORT, to use non-default ports for the master

# - SPARK\_MASTER\_OPTS, to set config properties only for the master (e.g. "-Dx=y")

# - SPARK\_WORKER\_CORES, to set the number of cores to use on this machine

# - SPARK\_WORKER\_MEMORY, to set how much total memory workers have to give executors (e.g. 1000m, 2g)

# - SPARK\_WORKER\_PORT / SPARK\_WORKER\_WEBUI\_PORT, to use non-default ports for the worker

# - SPARK\_WORKER\_INSTANCES, to set the number of worker processes per node

# - SPARK\_WORKER\_DIR, to set the working directory of worker processes

# - SPARK\_WORKER\_OPTS, to set config properties only for the worker (e.g. "-Dx=y")

# - SPARK\_DAEMON\_MEMORY, to allocate to the master, worker and history server themselves (default: 1g). # - SPARK\_HISTORY\_OPTS, to set config properties only for the history server (e.g. "-Dx=y")

# - SPARK\_SHUFFLE\_OPTS, to set config properties only for the external shuffle service (e.g. "-Dx=y")

# - SPARK\_DAEMON\_JAVA\_OPTS, to set config properties for all daemons (e.g. "-Dx=y")

# - SPARK\_PUBLIC\_DNS, to set the public dns name of the master or workers

# Generic options for the daemons used in the standalone deploy mode

# - SPARK\_CONF\_DIR Alternate conf dir. (Default: ${SPARK\_HOME}/conf)

# - SPARK\_LOG\_DIR Where log files are stored. (Default: ${SPARK\_HOME}/logs)

# - SPARK\_PID\_DIR Where the pid file is stored. (Default: /tmp)

# - SPARK\_IDENT\_STRING A string representing this instance of spark. (Default: $USER)

# - SPARK\_NICENESS The scheduling priority for daemons. (Default: 0)

# - SPARK\_NO\_DAEMONIZE Run the proposed command in the foreground. It will not output a PID file.

######################################################################################################### # Set MapR attributes and compute classpath

#########################################################################################################

# Set the spark attributes

if [ -d "/opt/mapr/spark/spark-2.1.0" ]; then export SPARK\_HOME=/opt/mapr/spark/spark-2.1.0 fi

# Load the hadoop version attributes

source /opt/mapr/spark/spark-2.1.0/mapr-util/hadoop-version-picker.sh export HADOOP\_HOME=$hadoop\_home\_dir export HADOOP\_CONF\_DIR=$hadoop\_conf\_dir

# Enable mapr impersonation export MAPR\_IMPERSONATION\_ENABLED=1

MAPR\_HADOOP\_CLASSPATH=`/opt/mapr/spark/spark-2.1.0/bin/mapr-classpath.sh`

MAPR\_HADOOP\_JNI\_PATH=`hadoop jnipath`

MAPR\_SPARK\_CLASSPATH="$MAPR\_HADOOP\_CLASSPATH"

SPARK\_MAPR\_HOME=/opt/mapr

export SPARK\_LIBRARY\_PATH=$MAPR\_HADOOP\_JNI\_PATH

export LD\_LIBRARY\_PATH="$MAPR\_HADOOP\_JNI\_PATH:$LD\_LIBRARY\_PATH"

# Load the classpath generator script

source /opt/mapr/spark/spark-2.1.0/mapr-util/generate-classpath.sh

# Calculate hive jars to include in classpath generate\_compatible\_classpath "spark" "2.1.0" "hive"

MAPR\_HIVE\_CLASSPATH=${generated\_classpath} if [ ! -z "$MAPR\_HIVE\_CLASSPATH" ]; then

MAPR\_SPARK\_CLASSPATH="$MAPR\_SPARK\_CLASSPATH:$MAPR\_HIVE\_CLASSPATH" fi

# Calculate hbase jars to include in classpath generate\_compatible\_classpath "spark" "2.1.0" "hbase"

MAPR\_HBASE\_CLASSPATH=${generated\_classpath} if [ ! -z "$MAPR\_HBASE\_CLASSPATH" ]; then

MAPR\_SPARK\_CLASSPATH="$MAPR\_SPARK\_CLASSPATH:$MAPR\_HBASE\_CLASSPATH"

SPARK\_SUBMIT\_OPTS="$SPARK\_SUBMIT\_OPTS -Dspark.driver.extraClassPath=$MAPR\_HBASE\_CLASSPATH" fi

# Set executor classpath for MESOS. Uncomment following string if you want deploy spark jobs on Mesos #MAPR\_MESOS\_CLASSPATH=$MAPR\_SPARK\_CLASSPATH

SPARK\_SUBMIT\_OPTS="$SPARK\_SUBMIT\_OPTS -Dspark.executor.extraClassPath=$MAPR\_HBASE\_CLASSPATH:$MAPR\_MESOS\_CLASSPATH"

# Set SPARK\_DIST\_CLASSPATH

export SPARK\_DIST\_CLASSPATH=$MAPR\_SPARK\_CLASSPATH

# Security status source /opt/mapr/conf/env.sh

if [ "$MAPR\_SECURITY\_STATUS" = "true" ]; then

SPARK\_SUBMIT\_OPTS="$SPARK\_SUBMIT\_OPTS -Dhadoop.login=hybrid -Dmapr\_sec\_enabled=true" fi

# scala

export SCALA\_VERSION=2.11

export SPARK\_SCALA\_VERSION=$SCALA\_VERSION export SCALA\_HOME=/opt/mapr/spark/spark-2.1.0/scala export SCALA\_LIBRARY\_PATH=$SCALA\_HOME/lib

# Use a fixed identifier for pid files export SPARK\_IDENT\_STRING="mapr"

######################################################################################################### # :::CAUTION::: DO NOT EDIT ANYTHING ON OR ABOVE THIS LINE

#########################################################################################################

#

# MASTER HA SETTINGS

#

#export SPARK\_DAEMON\_JAVA\_OPTS="-Dspark.deploy.recoveryMode=ZOOKEEPER -Dspark.deploy.zookeeper.url=<zookeerper1:5181,zookeep Djava.security.auth.login.config=/opt/mapr/conf/mapr.login.conf -Dzookeeper.sasl.client=false"

# MEMORY SETTINGS

export SPARK\_DAEMON\_MEMORY=1g export SPARK\_WORKER\_MEMORY=16g # Worker Directory

export SPARK\_WORKER\_DIR=$SPARK\_HOME/tmp

# Environment variable for printing spark command everytime you run spark.Set to "1" to print. # export SPARK\_PRINT\_LAUNCH\_COMMAND=1

/opt/mapr/spark/spark-2.2.1/bin

>>>>>>>>>>>>>>>>>>>>>>>>>>>> sparklyr<<<<<<<<<<<<<<<<<<<<<<<<<

Below are the instruction on how to install library for sparklyr incase asked in future.

* Go inside R as root.
* install.packages(“sparklyr”)
* choose 33 (for Sweden)
* it should work, incase dependencies not getting installed. We may need to do “yum install <>” for curl, xml, openssl, httr packages.

For “**sparklyr.nested**” library

* Go inside R as root.
* Install.packages(“devtools”)
* devtools::install\_github("mitre/sparklyr.nested")
* library(sparklyr.nested)

>>>>>>>>>>>>>>>> IMAPALA<<<<<<<<<<<<<<<<<<

Whenever we will get the any impala connection issue please execute the below command from **esekilxgp01.**

**clush -b -g prod "impala-shell -i localhost:21000 -q 'select 1 from bpn.summary'"**

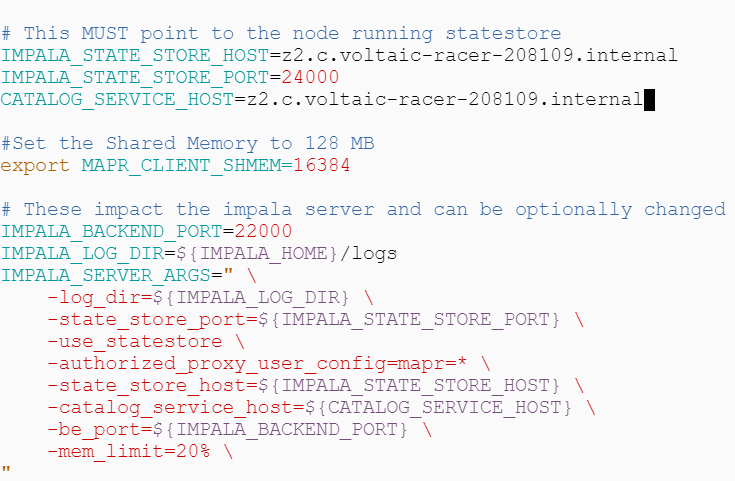
If you get “**clush: esekilx[5646-5647] (2): exited with exit code: 1**”, it means on these two node there is no running impalad daemon.

If you get “**clush: esekilx[5639-5640] (2): exited with exit code: 255”,** it means on these two node there are running impalad daemon, but it can’t fetch data from the impala. So in that case you need to debug the impalad daemon issue on those nodes.

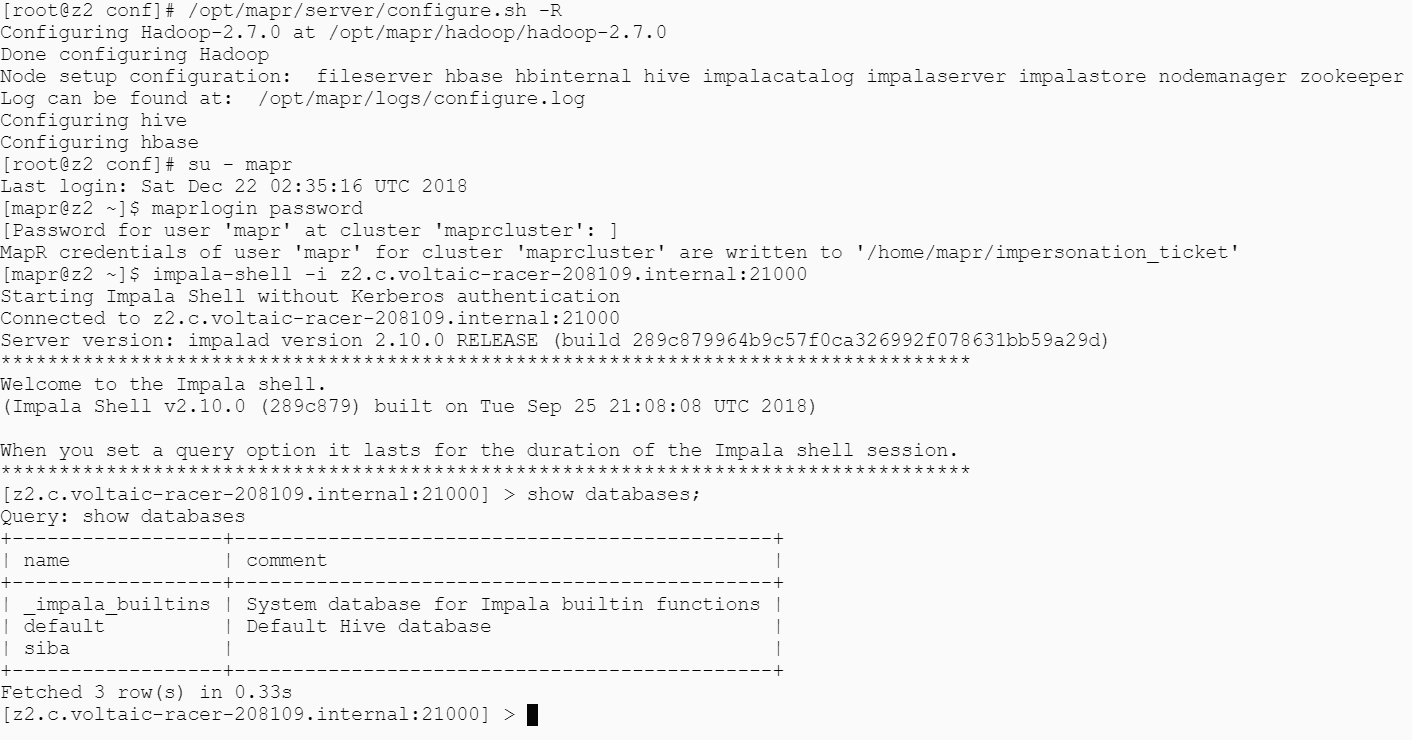
To test JDBC port, i.e port no: 21050 (HAPROXY port: 21051) you need to have any jdbc-database viewer like **DBeaver**.

**Imapal installation steps:**

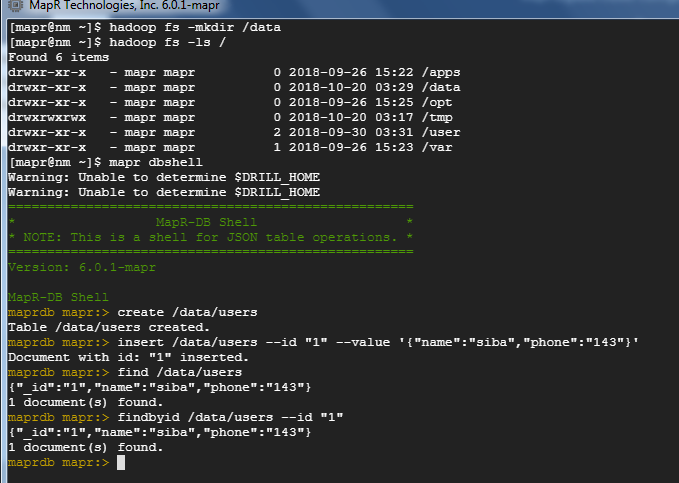
1. yum -y yum install mapr-impala mapr-impala-statestore mapr-impala-catalog mapr-impala-server
2. cd /opt/mapr/impala/impala-2.10.0/conf
3. cp /opt/mapr/hive/hive-2.3/conf/hive-site.xml .
4. chown mapr:mapr \*
5. vi env.sh



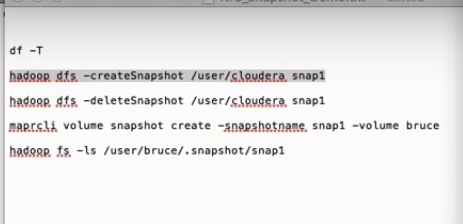
# /opt/mapr/server/configure.sh -R



**MapR-DB**



**Snapshot:**



for i in {3..6}; do ssh 10.142.0.$i "hostname; echo -e '\n'";scp -r cldb.key ssl\_keystore ssl\_truststore maprserverticket mapr@10.142.0.4:/tmp/;done

for i in {8..14}; do ssh 10.142.0.$i "hostname; echo -e '\n'";scp -r cldb.key ssl\_keystore ssl\_truststore maprserverticket mapr@10.142.0.$i:/tmp/;done

scp -r cldb.key ssl\_keystore ssl\_truststore maprserverticket mapr@10.142.0.4:/tmp/

watch -n 1 ls -ltr

for i in {2..6}; do ssh mapr@10.142.0.$i "hostname;jps ;echo -e '\n'";done >> test.out

for i in {8..14}; do ssh mapr@10.142.0.$i "hostname;sudo systemctl start mapr-zookeeper;sudo systemctl status mapr-zookeeper;sudo systemctl start mapr-warden;sudo systemctl status mapr-warden;echo -e '\n'";done >> service.out

rpm -qa | grep mapr | for i in `awk {'print$1'}`; do yum erase $i; done

rpm -qa | grep httpd | for i in `awk {'print$1'}`; do yum -y erase $i; done > test.txt

Vagrant

yum install gcc make kernel-devel -y

yum groupinstall " X Window System"

wget https://download.virtualbox.org/virtualbox/5.2.18/VirtualBox-5.2-5.2.18\_124319\_el7-1.x86\_64.rpm

wget https://releases.hashicorp.com/vagrant/2.1.4/vagrant\_2.1.4\_x86\_64.rpm

yum localinstall VirtualBox-5.2-5.2.18\_124319\_el7-1.x86\_64.rpm

yum localinstall vagrant\_2.1.4\_x86\_64.rpm

/sbin/vboxconfig

vagrant --version

mkdir -p /siba/centos

cd /siba/centos

vagrant box add centos/7

vagrant init centos/7

vim Vagrantfile

vagrant init centos/7

vim Vagrantfile

vagrant box add centos/7

vim Vagrantfile

vagrant box add centos/7

vagrant ssh

vagrant up

vim Vagrantfile

vagrant ssh-config

vagrant box list

vboxmanage stproperty machinefolder /siba/centos

vboxmanage setproperty machinefolder /siba/centos

vagrant up

@@@@@@@

#!/bin/bash

for i in `cat prod-list`

do echo "----$i----"

ssh $i cp /etc/resolv.conf /etc/resolv.conf.org

cat /root/resolv.conf | ssh root@$i 'cat > /etc/resolv.conf'

done

@@@@@@@@@@@@

[mapr@z2 bin]$ /opt/mapr/spark/spark-2.3.1/bin/run-example --master yarn --deploy-mode client SparkPi 10

./run-example --master yarn sql.hive.SparkHiveExample

cp -p log4j.properties.template log4j.properties

bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn ./examples/jars/spark-examples\_2.11-2.1.0-mapr-1710.jar 10

./spark-shell --master="yarn"

Imp:

./spark-submit --class org.apache.spark.examples.SparkPi --master yarn ./examples/jars/spark-examples\_2.11-2.1.0-mapr-1710.jar 10

[mapr@z2 bin]$ ./spark-submit --class org.apache.spark.examples.SparkPi --master yarn /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 10

/opt/mapr/spark/spark-2.3.1/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode cluster –-driver-memory 4g –-excutor-memory 2g –excutor-cores 1 /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 2

Runtime pass executor memory and cores:

/bin/spark-submit --class org.apache.spark.examples.SparkPi \

--master yarn \

--deploy-mode cluster \

--driver-memory 4g \

--executor-memory 2g \

--executor-cores 1 \

--queue thequeue \

lib/spark-examples\*.jar \

10

#######$$$$

./run-example --master yarn SparkPi 20

@@@@@@@@@@@@@ HAPROXY########

listen mariadb\_cluster 0.0.0.0:3030

## MariaDB balance leastconn - the cluster listening on port 3030.

mode tcp

balance leastconn

option httpchk

server db1 192.168.1.11:3306 check port 9200

server db2 192.168.1.12:3306 check port 9200

listen stats 0.0.0.0:9000

## HAProxy stats web gui running on port 9000 - username and password: test

mode http

stats enable

stats uri /stats

stats realm HAProxy\ Statistics

stats auth test:test

stats admin if TRUE

##########@@@@2222@@@@###############

cat haproxy.cfg

#---------------------------------------------------------------------

# Example configuration for a possible web application. See the

# full configuration options online.

#

# http://haproxy.1wt.eu/download/1.4/doc/configuration.txt

#

#---------------------------------------------------------------------

#---------------------------------------------------------------------

# Global settings

#---------------------------------------------------------------------

global

# to have these messages end up in /var/log/haproxy.log you will

# need to:

#

# 1) configure syslog to accept network log events. This is done

# by adding the '-r' option to the SYSLOGD\_OPTIONS in

# /etc/sysconfig/syslog

#

# 2) configure local2 events to go to the /var/log/haproxy.log

# file. A line like the following can be added to

# /etc/sysconfig/syslog

#

# local2.\* /var/log/haproxy.log

#

log 127.0.0.1 local2

chroot /var/lib/haproxy

pidfile /var/run/haproxy.pid

maxconn 4000

user haproxy

group haproxy

daemon

# turn on stats unix socket

stats socket /var/lib/haproxy/stats

#---------------------------------------------------------------------

# common defaults that all the 'listen' and 'backend' sections will

# use if not designated in their block

#---------------------------------------------------------------------

defaults

mode http

log global

option httplog

option dontlognull

option http-server-close

option forwardfor except 127.0.0.0/8

option redispatch

retries 6

timeout http-request 1m

timeout queue 1m

timeout connect 1m

timeout client 1m

timeout server 1m

timeout http-keep-alive 1m

timeout check 10s

maxconn 6000

#---------------------------------------------------------------------

# main frontend which proxys to the backends

#---------------------------------------------------------------------

frontend main \*:5000

acl url\_static path\_beg -i /static /images /javascript /styleshe ets

acl url\_static path\_end -i .jpg .gif .png .css .js

use\_backend static if url\_static

default\_backend app

#---------------------------------------------------------------------

# static backend for serving up images, stylesheets and such

#---------------------------------------------------------------------

backend static

balance roundrobin

server static 127.0.0.1:4331 check

#---------------------------------------------------------------------

# round robin balancing between the various backends

#---------------------------------------------------------------------

backend app

balance roundrobin

server app1 127.0.0.1:5001 check

server app2 127.0.0.1:5002 check

server app3 127.0.0.1:5003 check

server app4 127.0.0.1:5004 check

# ---------------------------------------------------------------------

# This is the setup for Impala. Impala client connect to load\_balancer\_host:2500 3.

# HAProxy will balance connections among the list of servers listed below.

# The list of Impalad is listening at port 21000 for beeswax (impala-shell) or o riginal ODBC driver.

# For JDBC or ODBC version 2.x driver, use port 21050 instead of 21000.

# ---------------------------------------------------------------------

listen impala :25001

mode tcp

option tcplog

balance leastconn

server impala\_ha\_odbc\_1 134.138.210.132:21000

server impala\_ha\_odbc\_2 134.138.210.133:21000

server impala\_ha\_odbc\_3 134.138.210.134:21000

server impala\_ha\_odbc\_4 134.138.210.135:21000

server impala\_ha\_odbc\_5 134.138.210.136:21000

server impala\_ha\_odbc\_6 134.138.210.137:21000

server impala\_ha\_odbc\_7 134.138.210.138:21000

server impala\_ha\_odbc\_8 134.138.210.139:21000

server impala\_ha\_odbc\_9 134.138.210.140:21000

server impala\_ha\_odbc\_10 134.138.210.141:21000

server impala\_ha\_odbc\_11 134.138.210.142:21000

server impala\_ha\_odbc\_12 134.138.210.143:21000

# server impala\_ha\_odbc\_13 134.138.210.144:21000

server impala\_ha\_odbc\_14 134.138.210.146:21000

server impala\_ha\_odbc\_15 134.138.210.147:21000

server impala\_ha\_odbc\_16 134.138.210.148:21000

server impala\_ha\_odbc\_17 134.138.210.149:21000

server impala\_ha\_odbc\_18 134.138.210.153:21000

server impala\_ha\_odbc\_19 134.138.210.154:21000

server impala\_ha\_odbc\_20 134.138.210.152:21000

server impala\_ha\_odbc\_21 134.138.210.155:21000

server impala\_ha\_odbc\_22 134.138.210.156:21000

server impala\_ha\_odbc\_23 134.138.210.157:21000

server impala\_ha\_odbc\_24 134.138.210.158:21000

server impala\_ha\_odbc\_25 134.138.210.151:21000

#server impala\_ha\_odbc\_14 134.138.210.145:21000 (server offline atm)

# ---------------------------------------------------------------------

# Setup for Hue or other JDBC-enabled applications.

# In particular, Hue requires sticky sessions.

# The application connects to load\_balancer\_host:21051, and HAProxy balances

# connections to the associated hosts, where Impala listens for JDBC

# requests on port 21050.

# ---------------------------------------------------------------------

listen impalajdbc :21051

mode tcp

option tcplog

balance source

server impala\_ha\_jdbc\_1 134.138.210.132:21050

server impala\_ha\_jdbc\_2 134.138.210.133:21050

server impala\_ha\_jdbc\_3 134.138.210.134:21050

server impala\_ha\_jdbc\_4 134.138.210.135:21050

server impala\_ha\_jdbc\_5 134.138.210.136:21050

server impala\_ha\_jdbc\_6 134.138.210.137:21050

server impala\_ha\_jdbc\_7 134.138.210.138:21050

server impala\_ha\_jdbc\_8 134.138.210.139:21050

server impala\_ha\_jdbc\_9 134.138.210.140:21050

server impala\_ha\_jdbc\_10 134.138.210.141:21050

server impala\_ha\_jdbc\_11 134.138.210.142:21050

server impala\_ha\_jdbc\_12 134.138.210.143:21050

# server impala\_ha\_jdbc\_13 134.138.210.144:21050

server impala\_ha\_jdbc\_14 134.138.210.146:21050

server impala\_ha\_jdbc\_15 134.138.210.147:21050

server impala\_ha\_jdbc\_16 134.138.210.148:21050

server impala\_ha\_jdbc\_17 134.138.210.149:21050

server impala\_ha\_jdbc\_18 134.138.210.153:21050

server impala\_ha\_jdbc\_19 134.138.210.154:21050

server impala\_ha\_jdbc\_20 134.138.210.152:21050

server impala\_ha\_jdbc\_21 134.138.210.155:21050

server impala\_ha\_jdbc\_22 134.138.210.156:21050

server impala\_ha\_jdbc\_23 134.138.210.157:21050

server impala\_ha\_jdbc\_24 134.138.210.158:21050

server impala\_ha\_jdbc\_25 134.138.210.151:21050

#server impala\_ha\_jdbc\_14 134.138.210.145:21050 (server offline atm)

sed -i -e 's/SELINUX=permissive/SELINUX=disabled/g' /etc/sysconfig/selinux

firewall-cmd --permanent --add-port=3306/tcp

firewall-cmd –reload

@@@@@@@@@@@@@ LUKENCRYPT########

#!/bin/bash

#yum -y install cryptsetup

#yum -y update device-mapper

# timestamp="$(date +%Y-%m-%d.%H:%M:%S)"

#mv -v /etc/crypto /etc/crypto."$timestamp"

mkdir -p /etc/crypto

chmod -R go-rw /etc/crypto

#mv -v /etc/crypttab /etc/crypttab."$timestamp"

#mv -v /opt/mapr/disks.txt /opt/mapr/disks.txt."$timestamp"

tr -dc '[:graph:]' < /dev/random | head -c "${1:-512}" > /etc/crypto/lukskey.bin

chmod go-rw /etc/crypto/lukskey.bin

disks="sda sdb sdc sdd sde sdf sdg sdh sdi sdj sdk sdl sdm sdn sdo sdp"

for f in $disks

do

# rm -fv /etc/crypto/"$f"-key.bin

# cryptsetup close luks-"$f"

cryptsetup --batch-mode --use-random luksFormat /dev/"$f" /etc/crypto/lukskey.bin

cryptsetup luksOpen /dev/"$f" luks-"$f" < /etc/crypto/lukskey.bin

echo luks-"$f" /dev/"$f" /etc/crypto/lukskey.bin >> /etc/crypttab

echo /dev/mapper/luks-"$f" >> /root/setup\_files/disks.txt

done

#echo "Backup files created..."

#ls -l {/etc/crypto."$timestamp",/etc/crypttab."$timestamp",/opt/mapr/disks.txt."$timestamp"}

**Troubleshooting Livy**

curl -X POST --data '{"proxyUser":"ehasbja","kind": "pyspark"}' -H "Content-Type: application/json" esekilxgp02.rnd.ki.sw.ericsson.se:8998/sessions

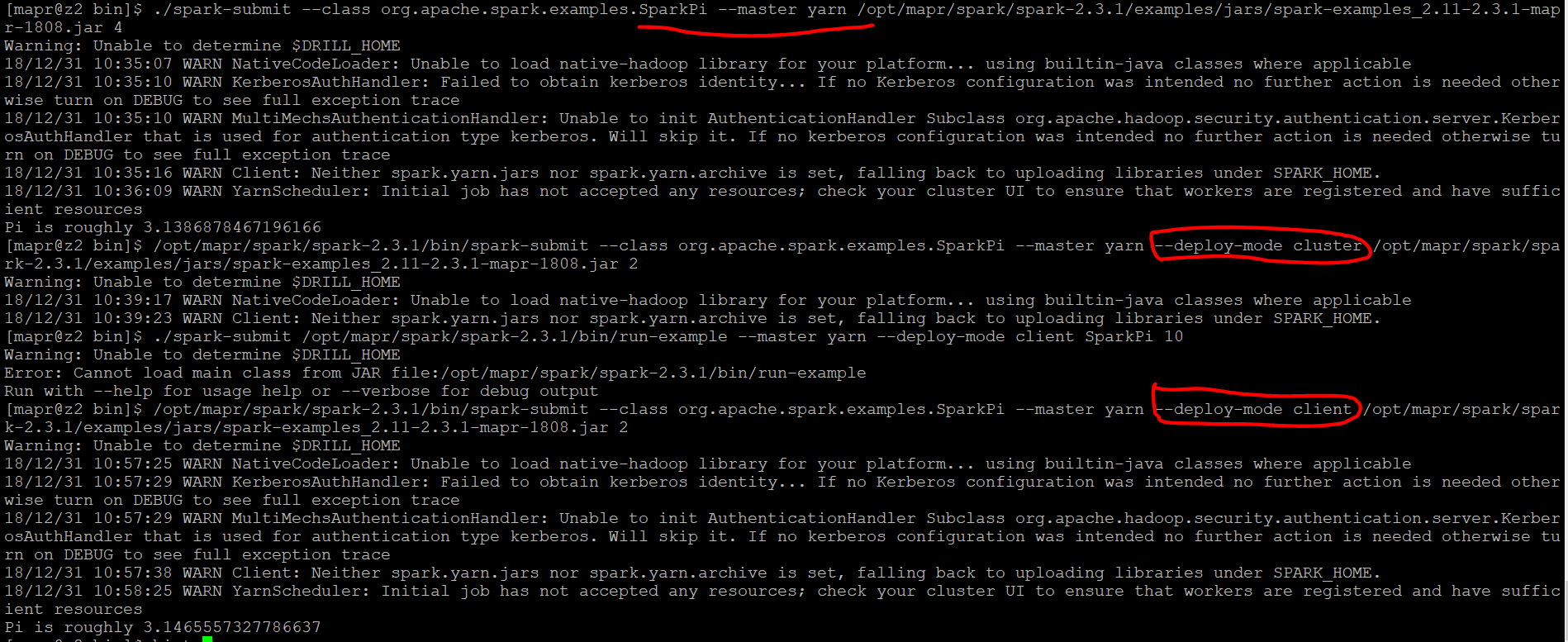
curl esekilxgp07.rnd.ki.sw.ericsson.se:8998/sessions/ | python -m json.tool

**Run Yarn Job**

[mapr@z2 bin]$ ./spark-submit --class org.apache.spark.examples.SparkPi --master yarn /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 10

[mapr@z2 bin]$ /opt/mapr/spark/spark-2.3.1/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode **cluster** /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 2

[mapr@z2 bin]$ /opt/mapr/spark/spark-2.3.1/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode **client** /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 2



/opt/mapr/spark/spark-2.3.1/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode cluster /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 2

[mapr@n5 hadoop-2.7.0]$ find . -name "\*emaples\*"

[mapr@n5 hadoop-2.7.0]$ find . -name "\*examples\*"

./share/hadoop/mapreduce/lib-examples

./share/hadoop/mapreduce/sources/hadoop-mapreduce-examples-2.7.0-mapr-1808-sources.jar

./share/hadoop/mapreduce/sources/hadoop-mapreduce-examples-2.7.0-mapr-1808-test-sources.jar

./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar

[mapr@n5 hadoop-2.7.0]$ yarn jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar

An example program must be given as the first argument.

Valid program names are:

aggregatewordcount: An Aggregate based map/reduce program that counts the words in the input files.

aggregatewordhist: An Aggregate based map/reduce program that computes the histogram of the words in the input files.

bbp: A map/reduce program that uses Bailey-Borwein-Plouffe to compute exact digits of Pi.

blocklocality: Checking Map job locality

dbcount: An example job that count the pageview counts from a database.

distbbp: A map/reduce program that uses a BBP-type formula to compute exact bits of Pi.

grep: A map/reduce program that counts the matches of a regex in the input.

join: A job that effects a join over sorted, equally partitioned datasets

multifilewc: A job that counts words from several files.

pentomino: A map/reduce tile laying program to find solutions to pentomino problems.

pi: A map/reduce program that estimates Pi using a quasi-Monte Carlo method.

randomtextwriter: A map/reduce program that writes 10GB of random textual data per node.

randomwriter: A map/reduce program that writes 10GB of random data per node.

secondarysort: An example defining a secondary sort to the reduce.

sleep: A job that sleeps at each map and reduce task.

sort: A map/reduce program that sorts the data written by the random writer.

sudoku: A sudoku solver.

terachecksum: Compute checksum of terasort output to compare with teragen checksum.

teragen: Generate data for the terasort

teragenwithcrc: Generate data for the terasort with CRC checksum

terasort: Run the terasort

terasortwithcrc: Run the terasort with CRC checksum

teravalidate: Checking results of terasort

teravalidaterecords: Checking results of terasort in terms of missing/duplicate records

teravalidatewithcrc: Checking results of terasort along with crc verification

wordcount: A map/reduce program that counts the words in the input files.

wordmean: A map/reduce program that counts the average length of the words in the input files.

wordmedian: A map/reduce program that counts the median length of the words in the input files.

wordstandarddeviation: A map/reduce program that counts the standard deviation of the length of the words in the input files.

[mapr@n5 hadoop-2.7.0]$ vi /tmp/a.txt

[mapr@n5 hadoop-2.7.0]$ hadoop fs -put /tmp/a.txt /tmp/a.txt

[mapr@n5 hadoop-2.7.0]$ ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt

bash: ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar: Permission denied

[mapr@n5 hadoop-2.7.0]$ vi /tmp/a.txt

[mapr@n5 hadoop-2.7.0]$ hadoop fs -put /tmp/a.txt /tmp/a.txt

[mapr@n5 hadoop-2.7.0]$ ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt

bash: ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar: Permission denied

[mapr@n5 hadoop-2.7.0]$ exit

[root@n5 conf]# cd /opt/mapr/hadoop/hadoop-2.7.0/

[root@n5 hadoop-2.7.0]# ll

[root@n5 hadoop-2.7.0]# ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt

-bash: ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar: Permission denied

[root@n5 hadoop-2.7.0]# su mapr

[mapr@n5 hadoop-2.7.0]$ maprlogin password

[Password for user 'mapr' at cluster 'maprcluster': ]

MapR credentials of user 'mapr' for cluster 'maprcluster' are written to '/tmp/maprticket\_5000'

[mapr@n5 hadoop-2.7.0]$ ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt

bash: ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0-mapr-1808.jar: Permission denied

[mapr@n5 hadoop-2.7.0]$ cd ./share/hadoop/mapreduce/

[mapr@n5 mapreduce]$ ls -lrt

[mapr@n5 mapreduce]$ chmod 777 hadoop-mapreduce-examples-2.7.0-mapr-1808.jar

chmod: changing permissions of ‘hadoop-mapreduce-examples-2.7.0-mapr-1808.jar’: Operation not permitted

[mapr@n5 mapreduce]$ exit

[root@n5 hadoop-2.7.0]# cd ./share/hadoop/mapreduce/

[root@n5 mapreduce]# chmod 777 hadoop-mapreduce-examples-2.7.0-mapr-1808.jar

[root@n5 mapreduce]# ./hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt

invalid file (bad magic number): Exec format error

[root@n5 mapreduce]# ./hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt /tmp/output

invalid file (bad magic number): Exec format error

[root@n5 mapreduce]# yarn jar /hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt /tmp/output

Not a valid JAR: /hadoop-mapreduce-examples-2.7.0-mapr-1808.jar

[root@n5 mapreduce]# yarn jar ./hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt /tmp/output

[root@n5 mapreduce]# su mapr

[mapr@n5 mapreduce]$ yarn jar ./hadoop-mapreduce-examples-2.7.0-mapr-1808.jar wordcount /tmp/a.txt /tmp/output

18/11/15 09:50:11 INFO client.RMProxy: Connecting to ResourceManager at n1.us-east1-b.c.main-form-217005.internal/10.142.0.3:8032

18/11/15 09:50:12 INFO input.FileInputFormat: Total input paths to process : 1

18/11/15 09:50:12 INFO mapreduce.JobSubmitter: number of splits:1

18/11/15 09:50:12 INFO mapreduce.JobSubmitter: Submitting tokens for job: job\_1542263789247\_0007

18/11/15 09:50:13 INFO security.ExternalTokenManagerFactory: Initialized external token manager class - com.mapr.hadoop.yarn.security.MapRTicketManager

18/11/15 09:50:13 INFO impl.YarnClientImpl: Submitted application application\_1542263789247\_0007

18/11/15 09:50:13 INFO mapreduce.Job: The url to track the job: https://n1.us-east1-b.c.main-form-217005.internal:8090/proxy/application\_1542263789247\_0007/

18/11/15 09:50:13 INFO mapreduce.Job: Running job: job\_1542263789247\_0007

18/11/15 09:50:25 INFO mapreduce.Job: Job job\_1542263789247\_0007 running in uber mode : false

18/11/15 09:50:25 INFO mapreduce.Job: map 0% reduce 0%

18/11/15 09:50:45 INFO mapreduce.Job: map 100% reduce 0%

18/11/15 09:50:58 INFO mapreduce.Job: map 100% reduce 100%

18/11/15 09:50:59 INFO mapreduce.Job: Job job\_1542263789247\_0007 completed successfully

18/11/15 09:50:59 INFO mapreduce.Job: Counters: 46

File System Counters

FILE: Number of bytes read=0

FILE: Number of bytes written=199163

FILE: Number of read operations=0

FILE: Number of large read operations=0

FILE: Number of write operations=0

MAPRFS: Number of bytes read=6482

MAPRFS: Number of bytes written=5596

MAPRFS: Number of read operations=595

MAPRFS: Number of large read operations=0

MAPRFS: Number of write operations=1605

Job Counters

Launched map tasks=1

MAPRFS: Number of bytes read=6482

MAPRFS: Number of bytes written=5596

MAPRFS: Number of read operations=595

MAPRFS: Number of large read operations=0

MAPRFS: Number of write operations=1605

Job Counters

Launched map tasks=1

Launched reduce tasks=1

Data-local map tasks=1

Total time spent by all maps in occupied slots (ms)=17773

Total time spent by all reduces in occupied slots (ms)=31101

Total time spent by all map tasks (ms)=17773

Total time spent by all reduce tasks (ms)=10367

Total vcore-seconds taken by all map tasks=17773

Total vcore-seconds taken by all reduce tasks=10367

Total megabyte-seconds taken by all map tasks=18199552

Total megabyte-seconds taken by all reduce tasks=31847424

DISK\_MILLIS\_MAPS=8887

DISK\_MILLIS\_REDUCES=13788

Map-Reduce Framework

Map input records=31

Map output records=336

Map output bytes=3560

Map output materialized bytes=0

Input split bytes=84

Combine input records=336

Combine output records=145

Reduce input groups=145

Reduce shuffle bytes=2044

Reduce input records=145

Reduce output records=145

Spilled Records=290

Shuffled Maps =1

Failed Shuffles=0

Merged Map outputs=2

GC time elapsed (ms)=292

CPU time spent (ms)=1010

Physical memory (bytes) snapshot=777093120

Virtual memory (bytes) snapshot=7567470592

Total committed heap usage (bytes)=1013710848

Shuffle Errors

IO\_ERROR=0

File Input Format Counters

Bytes Read=2274

File Output Format Counters

Bytes Written=1468

[mapr@n5 mapreduce]$

**$$$$$$$$$$$$$$$$$$###### Hue #####$$$$$$$$$$$$$**

**1.cp -r /usr/lib/python2.7/site-packages/google\_compute\_engine /opt/mapr/hue/hue-4.2.0/build/env/lib/python2.7/site-packages/**

**2.yum install mapr-hue**

Changed in the hue.init

[root@esekilx5638 conf]# diff --suppress-common-lines -y hue.ini hue.ini\_16\_11\_2018

hive\_server\_host=esekilx5636.rnd.ki.sw.ericsson.se | hive\_server\_host=esekilx5636.rnd.ki.sw.ericsson.se,esekilx5

#hive\_conf\_dir=/opt/mapr/hive/hive-2.1/conf | # hive\_conf\_dir=/opt/mapr/hive/hive-2.1

mechanism=none | mechanism=${mechanism}

server\_host=esekilx5634.rnd.ki.sw.ericsson.se | ## server\_host=localhost

server\_port=21051 | ## server\_port=21050

For google cloud:

cp -r /usr/lib/python2.7/site-packages/google\_compute\_engine /opt/mapr/hue/hue-4.2.0/build/env/lib/python2.7/site-packages/

**##########################dynamic allocation##############**

**#vi /opt/mapr/spark/spark-2.1.0/conf/spark-defaults.conf**

**# END OF THE SECURITY CONFIGURATION BLOCK**

**############################################**

**spark.dynamicAllocation.enabled true**

**spark.shuffle.service.enabled true**

**spark.dynamicAllocation.minExecutors 0**

**spark.executor.instances 0**

**spark.authenticate true**

**spark.executor.heartbeatInterval 1800s**

**spark.network.timeout 2400s**

**more spark-defaults.conf**

**# Default system properties included when running spark-submit.**

**# This is useful for setting default environmental settings.**

**# Log effective Spark configuration at startup on INFO level**

**spark.logConf true**

**# Enable event logs for HistoryServer**

**spark.eventLog.enabled true**

**spark.eventLog.dir maprfs:///apps/spark/logs**

**spark.history.fs.logDirectory maprfs:///apps/spark/logs**

**# Default location for Warehouse, if not using Hive**

**spark.sql.warehouse.dir /warehouse/spark**

**# Fix for SPARK-7819**

**spark.sql.hive.metastore.sharedPrefixes com.mysql.jdbc,org.postgresql,com.microsoft.sqlserver,oracle.jdbc,com.mapr.fs.shim.LibraryLoader,com.mapr.security.**

**JNISecurity,com.mapr.fs.jni,com.mapr.fs.ShimLoader**

**spark.executor.memory 2g**

**spark.yarn.archive maprfs:///apps/spark/spark-jars.zip**

**spark.history.ui.port 18080**

**# SECURITY BLOCK**

**# ALL SECURITY PROPERTIES MUST BE PLACED IN THIS BLOCKG**

**# Security**

**# - ACLS**

**spark.acls.enable true**

**spark.admin.acls mapr**

**spark.admin.acls.groups maprg**

**spark.authenticate.secret changeMe**

**spark.authenticate true**

**spark.ssl.enabled true**

**spark.io.encryption.enabled true**

**spark.io.encryption.keySizeBits 128**

**spark.ssl.fs.enabled true**

**spark.ssl.keyPassword mapr123**

**spark.ssl.keyStore /opt/mapr/conf/ssl\_keystore**

**spark.ssl.keyStorePassword mapr123**

**spark.ssl.trustStore /opt/mapr/conf/ssl\_truststore**

**spark.ssl.trustStorePassword mapr123**

**spark.ssl.protocol tls**

**spark.ssl.enabledAlgorithms TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA,TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA**

**spark.authenticate.enableSaslEncryption true**

**spark.network.sasl.serverAlwaysEncrypt true**

**##########################dynamic allocation##############**

**# END OF THE SECURITY CONFIGURATION BLOCK**

**############################################**

**spark.dynamicAllocation.enabled true**

**spark.shuffle.service.enabled true**

**spark.dynamicAllocation.minExecutors 0**

**spark.executor.instances 0**

**spark.authenticate true**

**spark.executor.heartbeatInterval 1800s**

**spark.network.timeout 2400s**

**[root@esekilxgp01 ~/dynamic\_alloction]# clush -b -g dev 'cp -p /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-env.sh /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-env.sh.14NOV2018.bak'**

**[root@esekilxgp01 ~/dynamic\_alloction]# clush -b -g dev --copy yarn-env.sh --dest /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/**

**[root@esekilxgp01 ~/dynamic\_alloction]# clush -b -g dev 'cp -p /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-site.xml /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-site.xml.14NOV2018.bak'**

**[root@esekilxgp01 ~/dynamic\_alloction]# clush -b -g dev --copy yarn-site.xml --dest /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/**

**clush -b -g dev 'ls -lrt /opt/mapr/spark/spark-2.1.0/yarn/spark-2.1.0-mapr-1707-yarn-shuffle.jar'**

**clush -b -g dev 'cp /opt/mapr/spark/spark-2.1.0/yarn/spark-2.1.0-mapr-1710-yarn-shuffle.jar /opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/yarn/lib/'**

**$$$$$$$$#######yarn-site.xml########$$$$$**

**<property>**

**<name>yarn.nodemanager.aux-services</name>**

**<value>mapreduce\_shuffle,mapr\_direct\_shuffle,spark\_shuffle</value>**

**</property>**

**<property>**

**<name>yarn.nodemanager.aux-services.spark\_shuffle.class</name>**

**<value>org.apache.spark.network.yarn.YarnShuffleService</value>**

**</property>**

**<property>**

**<name>spark.authenticate</name>**

**<value>true</value>**

**</property>**

**--**

<property>

<name>yarn.resourcemanager.webapp.address</name>

<value>8088</value>

</property>

#/opt/mapr/spark/spark-2.3.1/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode cluster /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 2

**# yarn logs -applicationId application\_1638783255\_7394792 > /tmp/application\_1638783255\_7394792.log**

**# grep -i spark-2.1.0-mapr-1710-yarn-shuffle.jar /tmp/application\_1638783255\_7394792.log**

**Mapr loglive ticket**

maprlogin generateticket -type service -out /tmp/long\_ticket -duration 3650:0:0 -renewal 3650:0:0 -user mapr

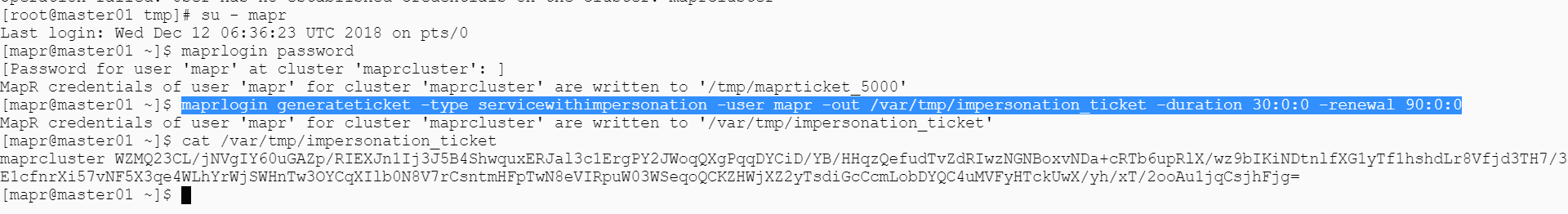
maprlogin generateticket -type service -out /tmp/long\_ticket -duration 3650:0:0 -renewal 3650:0:0 -user mapr

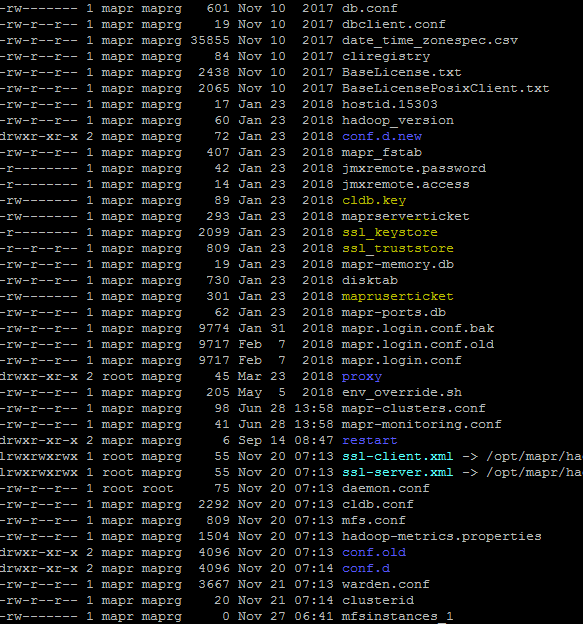
maprlogin generateticket -type servicewithimpersonation -user enaysib -out /var/tmp/impersonation\_ticket -duration 9999:0:0 -renewal

or

maprlogin generateticket -type servicewithimpersonation -user mapr -out /var/tmp/impersonation\_ticket -duration 30:0:0 -renewal 90:0:0

https://mapr.com/docs/52/SecurityGuide/GeneratingServiceWithImpersonationTicket.html





OPentstb

**OTSDB-72**

**Issue:** The memory allocated to OpenTSDB can be insufficient, resulting in empty graphs and out-of-memory or GC overhead limit exceeded errors.

**Workaround:** Increase the default memory for OpenTSDB by making the following changes on all OpenTSDB nodes:

1. Edit the /opt/mapr/conf/conf.d/warden.opentsdb.conf file to change:
2. service.heapsize.max=2000

service.heapsize.min=2000

to

service.heapsize.max=6000

service.heapsize.min=6000

1. Edit the /opt/mapr/opentsdb/opentsdb-\*/etc/init.d/opentsdb file to change:
2. $

{JVMXMX:=-Xmx2000m -Xss1m -XX:MaxMetaspaceSize=128m}

to

$

{JVMXMX:=-Xmx6000m -Xss1m -XX:MaxMetaspaceSize=128m}

1. Restart the OpenTSDB service:

maprcli node services -name opentsdb -nodes <space-separated list of OpenTSDB nodes> -action restart

#################Hue #################

1.cp -r /usr/lib/python2.7/site-packages/google\_compute\_engine /opt/mapr/hue/hue-4.2.0/build/env/lib/python2.7/site-packages/

2.yum install mapr-hue

3. cd /opt/mapr/hue/hue-4.2.0/desktop/conf

vi hue.ini

4./opt/mapr/server/configure.sh -R

**----------R and sparkR and Spark and hive** ---------------

[root@esekilxgp05 ~]# yum install R

[root@esekilxgp05 ~]# which R

Set $HOME in /usr/bin and its show in env

[root@esekilxgp05 ~]# cat .bash\_profile

# .bash\_profile

e

# Get the aliases and functions if [ -f ~/.bashrc ]; then . ~/.bashrc fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH

[root@esekilxgp05 ~]# env

PATH=/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/usr/java/jdk1.8.0\_161//bin:/usr/java/

**For sparkR and Spark and hive--------------------**

[root@esekilxgp05 ~]# yum install -y mapr-spark mapr-hive

00000

yum install mapr-spark mapr-spark-master mapr-spark-historyserver mapr-spark-thriftserver

[mapr@n2 bin]$ hadoop fs -mkdir /apps/spark

[mapr@n2 bin]$ hadoop fs -chmod 777 /apps/spark

[mapr@n2 bin]$ logout

[root@n2 conf]# /opt/mapr/server/configure.sh -R

00000

[root@esekilxgp05 ~]# cd /opt/mapr/spark/spark-2.1.0/conf/

[root@esekilxgp05 ~]#ls -ltar

-rwxr-xr-x 1 mapr maprg 7191 Oct 11 07:51 spark-env.sh

-rw-r--r-- 1 mapr maprg 3727 Oct 11 07:51 hive-site.xml

-rw-r--r-- 1 mapr maprg 1749 Oct 11 07:52 spark-defaults.conf

[root@esekilxgp05 ~]# vi spark-defaults.conf

Add below files:

spark.yarn.dist.files = /opt/mapr/spark/spark-2.1.0/conf/hive-site.xml

spark.sql.hive.metastore.version = 1.2.0

[root@esekilxgp05 conf]# cat hive-site.xml

<?xml version="1.0"?>

<!--

Licensed to the Apache Software Foundation (ASF) under one or more contributor license agreements. See the NOTICE file distributed with this work for additional information regarding copyright ownership. The ASF licenses this file to you under the Apache License, Version 2.0 (the

"License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

-->

<configuration>

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://esekilxgp06.rnd.ki.sw.ericsson.se:3306/hive?createDatabaseIfNotExist=true</value> <description>JDBC connect string for a JDBC metastore</description>

</property>

<property>

<name>javax.jdo.option.ConnectionDriverName</name>

<value>com.mysql.jdbc.Driver</value>

<description>Driver class name for a JDBC metastore</description> </property>

<property>

<name>javax.jdo.option.ConnectionUserName</name>

<value>hive</value>

<description>username to use against metastore database</description> </property>

<property>

<name>javax.jdo.option.ConnectionPassword</name>

<value>CyT4cPPfwDs</value>

<description>password to use against metastore database</description>

</property>

<property>

<name>hive.metastore.schema.verification</name>

<value>false</value>

</property>

<property>

<name>hive.metastore.uris</name>

<value>thrift://esekilx5636.rnd.ki.sw.ericsson.se:9083,thrift://esekilx5637.rnd.ki.sw.ericsson.se:9083</value>

</property>

<!-- For hive server2 -->

<property>

<name>hive.server2.enable.doAs</name>

<value>true</value>

</property>

<!-- For hive server2 and meta store -->

<property>

<name>hive.metastore.execute.setugi</name>

<value>true</value>

</property>

<property>

<name>hive.metastore.warehouse.dir</name>

<value>/project/rdi/warehouse/hive</value>

<description>location of default database for the warehouse</description> </property>

<property>

<name>hive.metastore.try.direct.sql</name>

<value>true</value>

<description>

Whether the Hive metastore should try to use direct SQL queries instead of the

DataNucleus for certain read paths. This can improve metastore performance when fetching many partitions or column statistics by orders of magnitude; however, it is not guaranteed to work on all RDBMS-es and all versions. In case of SQL failures, the metastore will fall back to the DataNucleus, so it's safe even if SQL doesn't work for all queries on your datastore. If all SQL queries fail (for example, your metastore is backed by MongoDB), you might want to disable this to save the try-and-fall-back cost.

</description>

</property>

<property>

<name>hive.metastore.client.socket.timeout</name>

<value>1800s</value>

<description>

Expects a time value with unit (d/day, h/hour, m/min, s/sec, ms/msec, us/usec, ns/nsec), which is sec if not specified.

MetaStore Client socket timeout in seconds

</description>

</property>

<property>

<name>hive.metastore.sasl.enabled </name>

<value>false</value>

</property>

</configuration>

[root@esekilxgp05 conf]# cat spark-env.sh

#!/usr/bin/env bash

#

# Licensed to the Apache Software Foundation (ASF) under one or more

# contributor license agreements. See the NOTICE file distributed with # this work for additional information regarding copyright ownership.

# The ASF licenses this file to You under the Apache License, Version 2.0

# (the "License"); you may not use this file except in compliance with

# the License. You may obtain a copy of the License at #

# http://www.apache.org/licenses/LICENSE-2.0

#

# Unless required by applicable law or agreed to in writing, software

# distributed under the License is distributed on an "AS IS" BASIS,

# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. # See the License for the specific language governing permissions and # limitations under the License.

#

# This file is sourced when running various Spark programs.

# Copy it as spark-env.sh and edit that to configure Spark for your site.

# Options read when launching programs locally with

# ./bin/run-example or ./bin/spark-submit

# - HADOOP\_CONF\_DIR, to point Spark towards Hadoop configuration files

# - SPARK\_LOCAL\_IP, to set the IP address Spark binds to on this node

# - SPARK\_PUBLIC\_DNS, to set the public dns name of the driver program # - SPARK\_CLASSPATH, default classpath entries to append

# Options read by executors and drivers running inside the cluster

# - SPARK\_LOCAL\_IP, to set the IP address Spark binds to on this node

# - SPARK\_PUBLIC\_DNS, to set the public DNS name of the driver program

# - SPARK\_CLASSPATH, default classpath entries to append

# - SPARK\_LOCAL\_DIRS, storage directories to use on this node for shuffle and RDD data # - MESOS\_NATIVE\_JAVA\_LIBRARY, to point to your libmesos.so if you use Mesos

# Options read in YARN client mode

# - HADOOP\_CONF\_DIR, to point Spark towards Hadoop configuration files

# - SPARK\_EXECUTOR\_INSTANCES, Number of executors to start (Default: 2) # - SPARK\_EXECUTOR\_CORES, Number of cores for the executors (Default: 1).

# - SPARK\_EXECUTOR\_MEMORY, Memory per Executor (e.g. 1000M, 2G) (Default: 1G)

# - SPARK\_DRIVER\_MEMORY, Memory for Driver (e.g. 1000M, 2G) (Default: 1G)

# Options for the daemons used in the standalone deploy mode

# - SPARK\_MASTER\_HOST, to bind the master to a different IP address or hostname

# - SPARK\_MASTER\_PORT / SPARK\_MASTER\_WEBUI\_PORT, to use non-default ports for the master

# - SPARK\_MASTER\_OPTS, to set config properties only for the master (e.g. "-Dx=y")

# - SPARK\_WORKER\_CORES, to set the number of cores to use on this machine

# - SPARK\_WORKER\_MEMORY, to set how much total memory workers have to give executors (e.g. 1000m, 2g)

# - SPARK\_WORKER\_PORT / SPARK\_WORKER\_WEBUI\_PORT, to use non-default ports for the worker

# - SPARK\_WORKER\_INSTANCES, to set the number of worker processes per node

# - SPARK\_WORKER\_DIR, to set the working directory of worker processes

# - SPARK\_WORKER\_OPTS, to set config properties only for the worker (e.g. "-Dx=y")

# - SPARK\_DAEMON\_MEMORY, to allocate to the master, worker and history server themselves (default: 1g). # - SPARK\_HISTORY\_OPTS, to set config properties only for the history server (e.g. "-Dx=y")

# - SPARK\_SHUFFLE\_OPTS, to set config properties only for the external shuffle service (e.g. "-Dx=y")

# - SPARK\_DAEMON\_JAVA\_OPTS, to set config properties for all daemons (e.g. "-Dx=y")

# - SPARK\_PUBLIC\_DNS, to set the public dns name of the master or workers

# Generic options for the daemons used in the standalone deploy mode

# - SPARK\_CONF\_DIR Alternate conf dir. (Default: ${SPARK\_HOME}/conf)

# - SPARK\_LOG\_DIR Where log files are stored. (Default: ${SPARK\_HOME}/logs)

# - SPARK\_PID\_DIR Where the pid file is stored. (Default: /tmp)

# - SPARK\_IDENT\_STRING A string representing this instance of spark. (Default: $USER)

# - SPARK\_NICENESS The scheduling priority for daemons. (Default: 0)

# - SPARK\_NO\_DAEMONIZE Run the proposed command in the foreground. It will not output a PID file.

######################################################################################################### # Set MapR attributes and compute classpath

#########################################################################################################

# Set the spark attributes

if [ -d "/opt/mapr/spark/spark-2.1.0" ]; then export SPARK\_HOME=/opt/mapr/spark/spark-2.1.0 fi

# Load the hadoop version attributes

source /opt/mapr/spark/spark-2.1.0/mapr-util/hadoop-version-picker.sh export HADOOP\_HOME=$hadoop\_home\_dir export HADOOP\_CONF\_DIR=$hadoop\_conf\_dir

# Enable mapr impersonation export MAPR\_IMPERSONATION\_ENABLED=1

MAPR\_HADOOP\_CLASSPATH=`/opt/mapr/spark/spark-2.1.0/bin/mapr-classpath.sh`

MAPR\_HADOOP\_JNI\_PATH=`hadoop jnipath`

MAPR\_SPARK\_CLASSPATH="$MAPR\_HADOOP\_CLASSPATH"

SPARK\_MAPR\_HOME=/opt/mapr

export SPARK\_LIBRARY\_PATH=$MAPR\_HADOOP\_JNI\_PATH

export LD\_LIBRARY\_PATH="$MAPR\_HADOOP\_JNI\_PATH:$LD\_LIBRARY\_PATH"

# Load the classpath generator script

source /opt/mapr/spark/spark-2.1.0/mapr-util/generate-classpath.sh

# Calculate hive jars to include in classpath generate\_compatible\_classpath "spark" "2.1.0" "hive"

MAPR\_HIVE\_CLASSPATH=${generated\_classpath} if [ ! -z "$MAPR\_HIVE\_CLASSPATH" ]; then

MAPR\_SPARK\_CLASSPATH="$MAPR\_SPARK\_CLASSPATH:$MAPR\_HIVE\_CLASSPATH" fi

# Calculate hbase jars to include in classpath generate\_compatible\_classpath "spark" "2.1.0" "hbase"

MAPR\_HBASE\_CLASSPATH=${generated\_classpath} if [ ! -z "$MAPR\_HBASE\_CLASSPATH" ]; then

MAPR\_SPARK\_CLASSPATH="$MAPR\_SPARK\_CLASSPATH:$MAPR\_HBASE\_CLASSPATH"

SPARK\_SUBMIT\_OPTS="$SPARK\_SUBMIT\_OPTS -Dspark.driver.extraClassPath=$MAPR\_HBASE\_CLASSPATH" fi

# Set executor classpath for MESOS. Uncomment following string if you want deploy spark jobs on Mesos #MAPR\_MESOS\_CLASSPATH=$MAPR\_SPARK\_CLASSPATH

SPARK\_SUBMIT\_OPTS="$SPARK\_SUBMIT\_OPTS -Dspark.executor.extraClassPath=$MAPR\_HBASE\_CLASSPATH:$MAPR\_MESOS\_CLASSPATH"

# Set SPARK\_DIST\_CLASSPATH

export SPARK\_DIST\_CLASSPATH=$MAPR\_SPARK\_CLASSPATH

# Security status source /opt/mapr/conf/env.sh

if [ "$MAPR\_SECURITY\_STATUS" = "true" ]; then

SPARK\_SUBMIT\_OPTS="$SPARK\_SUBMIT\_OPTS -Dhadoop.login=hybrid -Dmapr\_sec\_enabled=true" fi

# scala

export SCALA\_VERSION=2.11

export SPARK\_SCALA\_VERSION=$SCALA\_VERSION export SCALA\_HOME=/opt/mapr/spark/spark-2.1.0/scala export SCALA\_LIBRARY\_PATH=$SCALA\_HOME/lib

# Use a fixed identifier for pid files export SPARK\_IDENT\_STRING="mapr"

######################################################################################################### # :::CAUTION::: DO NOT EDIT ANYTHING ON OR ABOVE THIS LINE

#########################################################################################################

#

# MASTER HA SETTINGS

#

#export SPARK\_DAEMON\_JAVA\_OPTS="-Dspark.deploy.recoveryMode=ZOOKEEPER -Dspark.deploy.zookeeper.url=<zookeerper1:5181,zookeep Djava.security.auth.login.config=/opt/mapr/conf/mapr.login.conf -Dzookeeper.sasl.client=false"

# MEMORY SETTINGS

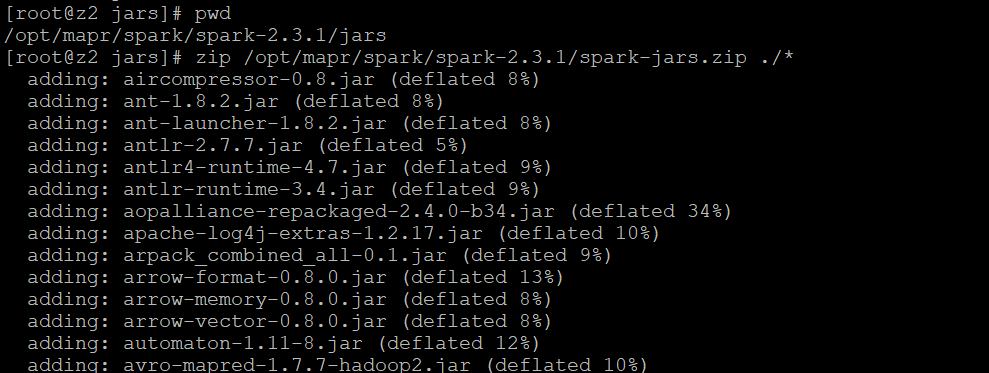
export SPARK\_DAEMON\_MEMORY=1g export SPARK\_WORKER\_MEMORY=16g # Worker Directory

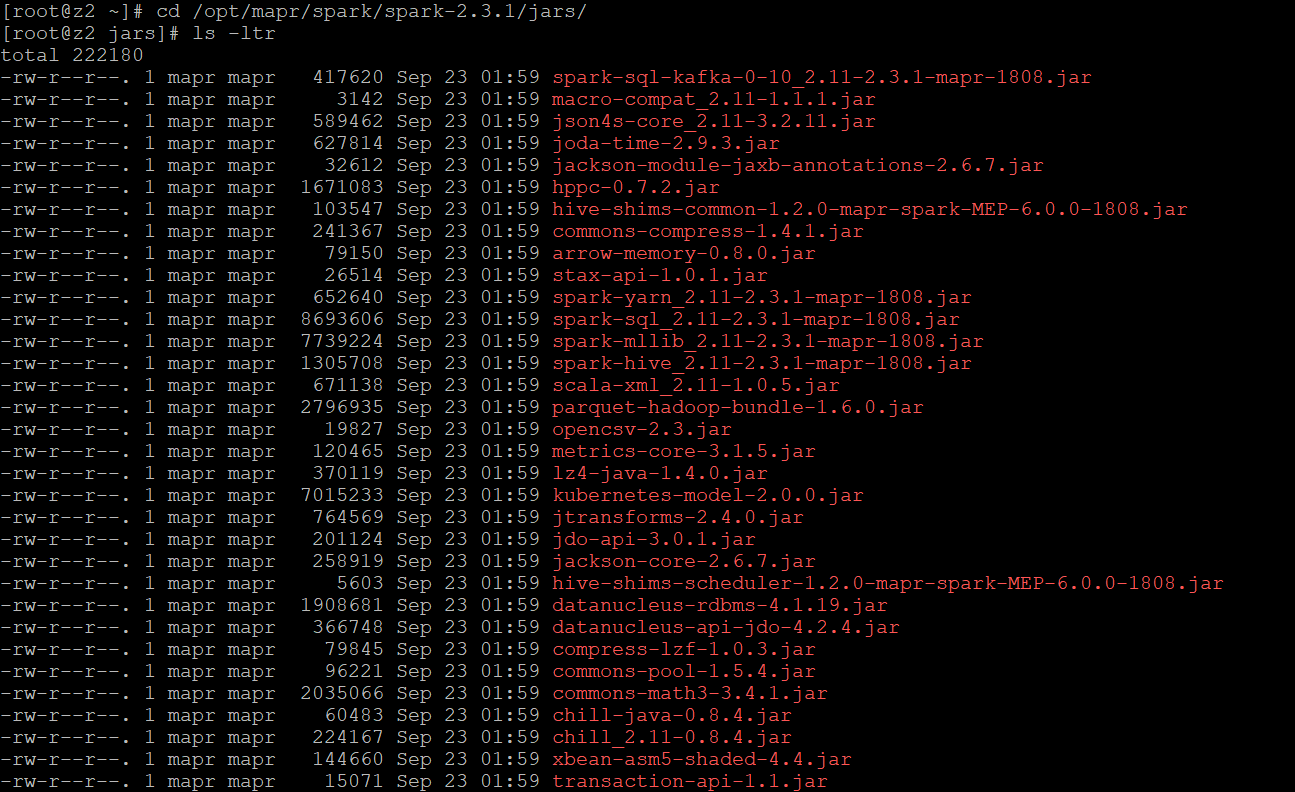
export SPARK\_WORKER\_DIR=$SPARK\_HOME/tmp

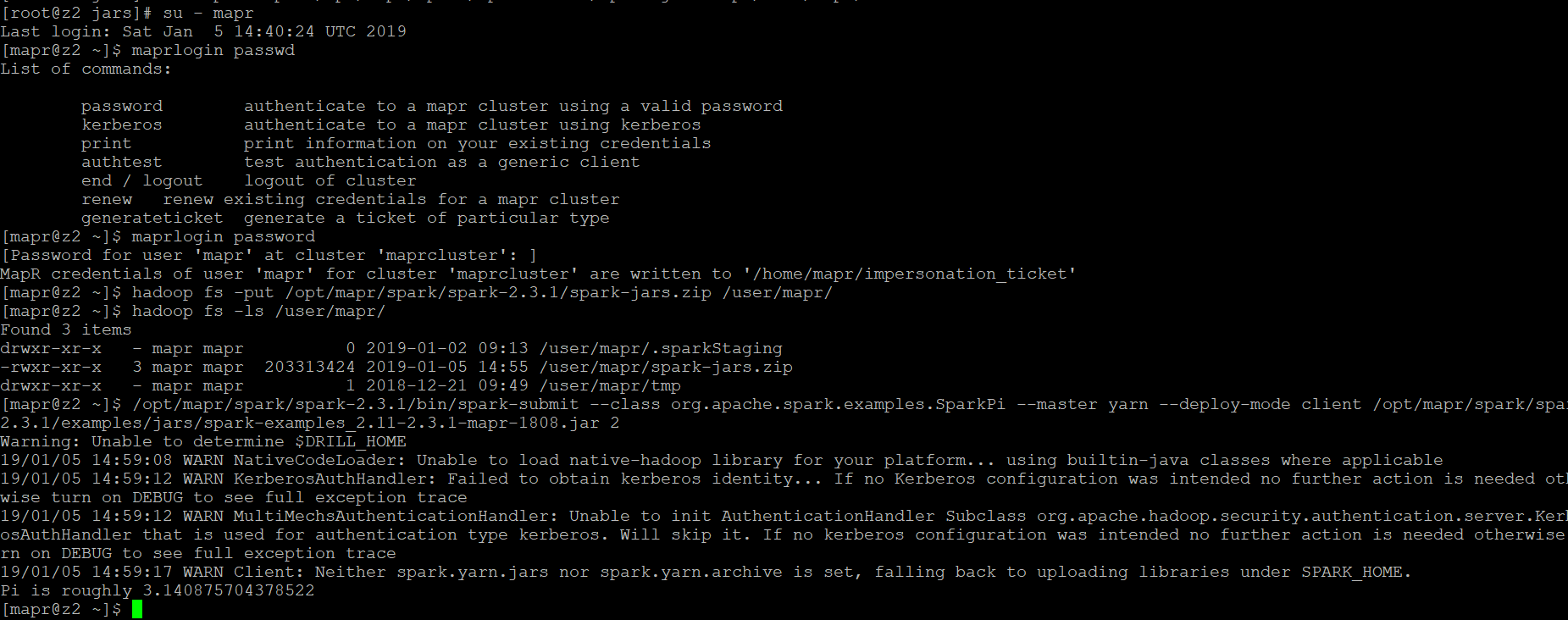
# Environment variable for printing spark command everytime you run spark.Set to "1" to print. # export SPARK\_PRINT\_LAUNCH\_COMMAND=1

/opt/mapr/spark/spark-2.2.1/bin

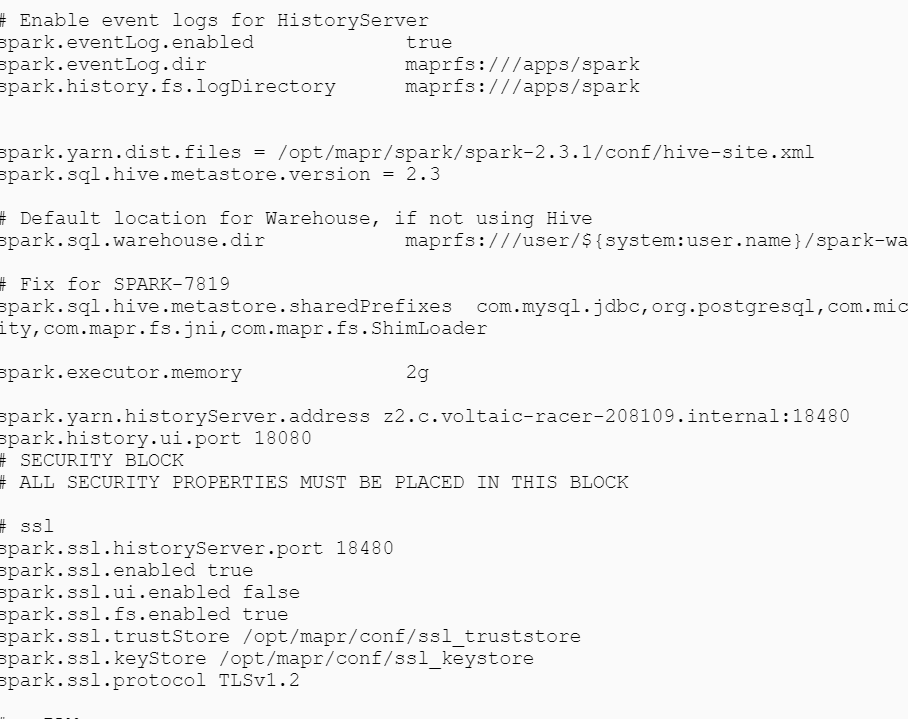
http://35.227.126.16:4040











**>>>>>>>>>>>>>>>>>>>>>>>>>>>> sparklyr<<<<<<<<<<<<<<<<<<<<<<<<<**

Below are the instruction on how to install library for sparklyr incase asked in future.

* Go inside R as root.
* install.packages(“sparklyr”)
* choose 33 (for Sweden)
* it should work, incase dependencies not getting installed. We may need to do “yum install <>” for curl, xml, openssl, httr packages.

For “**sparklyr.nested**” library

* Go inside R as root.
* Install.packages(“devtools”)
* devtools::install\_github("mitre/sparklyr.nested")
* library(sparklyr.nested)

**[root@z2 ~]# /opt/mapr/server/configure.sh -secure -Z master01.c.voltaic-racer-208109.internal:5181,z1.c.voltaic-racer-208109.internal:5181,z2.c.voltaic-racer-208109.internal:5181 -C master01.c.voltaic-racer-208109.internal:7222 -RM master01.c.voltaic-racer-208109.internal,z1.c.voltaic-racer-208109.internal -HS z1.c.voltaic-racer-208109.internal -N maprcluster -u mapr -g mapr -noDB**

**Disk Failure**

**Troubleshooting Steps: Disk Failure**

udevadm control --reload-rules && udevadm trigger

**1> Check disk status**

**Command :maprcli disk list -host 15.102.58.99**

**2> Check storage pool status**

**/opt/mapr/server/mrconfig sp list**

**if sp is offline then check which sp is offline and see the disk status of that sp.**

**3> If disk is failed then Check current disk state.(This command will show you the current state of the block device)**

**Command: cat /sys/block/<disk>/device/state**

**The required output will be: “running”**

**running**

**4> Check the “DMESG” command for issues on a specific disk.(This command is ideally helpful to determine whether there is an “I/O error” on any disks or any activity on the system generally.)**

**Command dmesg | grep <disk>**

**example: dmesg | grep -i sdd**

**sd 6:0:0:3: [sdd] Unhandled error code**

**sd 6:0:0:3: [sdd] Result: hostbyte=DID\_OK driverbyte=DRIVER\_TIMEOUT**

**sd 6:0:0:3: [sdd] CDB: Write(10): 2a 00 5e a2 9e 00 00 04 00 00**

**sd 6:0:0:3: [sdd] killing request**

**sd 6:0:0:3: [sdd] Unhandled error code**

**end\_request: I/O error, dev sdd, sector 1587712512**

**end\_request: I/O error, dev sdd, sector 1587711488**

**end\_request: I/O error, dev sdd, sector 1587710464**

**end\_request: I/O error, dev sdd, sector 1587709440**

**end\_request: I/O error, dev sdd, sector 1587708416**

**sd 6:0:0:3: [sdd] Result: hostbyte=DID\_NO\_CONNECT driverbyte=DRIVER\_OK**

**sd 6:0:0:3: [sdd] CDB: Read(10): 28 00 cf e2 64 40 00 00 10 00**

**end\_request: I/O error, dev sdd, sector 3487720512**

**5>If there is a significant error such as “I/O error” or the state of the disks are not showing “running”, please do contact to Linux(Hardware Team) for more verification. Further checks will be done from UX/LX team.**

**6>If this appears to be a software failure, ask them to replace it with a new disk.**

**7> after replacement,Run the command "maprcli disk remove -host 127.0.0.1 -disks /dev/sdd" to remove /dev/sdd from MapR-FS.(better do it from MCS)**

**8> In addition to /dev/sdd, the above command removes all the disks that belong to the same storage pool, from MapR-FS.**

**(Note down the names of all removed disks.)**

**9> Add all the above removed disks (exclude /dev/sdd) and the new disk to MapR-FS by running the command:**

**(if the disk added by EITTE it will be a raw disk so, use the following)**

**cryptsetup --batch-mode --use-random luksFormat /dev/sdg /etc/crypto/lukskey.bin**

**cryptsetup luksOpen /dev/sdg luks-sdg < /etc/crypto/lukskey.bin**

**Command "maprcli disk add -host 127.0.0.1 -disks <comma separated list of disks>"**

**If there is no new disk, the command would just be:**

**"maprcli disk add -host 127.0.0.1 -disks /dev/sdy,/dev/sdz"**

**/opt/mapr/server/disksetup -F -W 4 disks.txt**

**/opt/mapr/server/mrconfig sp list**

**/opt/mapr/server/mrconfig sp list -v**

**10> Stop and Start the warden**

**Collectd**

1. Install the following rpms for open source collectd (RPM can be downloaded from https://pkg.ci.collectd.org/rpm/collectd-5.8/epel-7-x86\_64/)

collectd-5.8.0.74.g0c85475-8.el7.centos.x86\_64.rpm

collectd-write\_prometheus-5.8.0.74.g0c85475-8.el7.centos.x86\_64.rpm

collectd-rrdtool-5.8.0.74.g0c85475-8.el7.centos.x86\_64.rpm

collectd-disk-5.8.0.74.g0c85475-8.el7.centos.x86\_64.rpm

2.Open collectd configuration file.

vi /etc/collectd.conf

3.Make sure that the following plugins are enabled in LoadPlugin section:

LoadPlugin cpu

LoadPlugin df

LoadPlugin disk

LoadPlugin interface

LoadPlugin load

LoadPlugin memory

LoadPlugin uptime

LoadPlugin users

LoadPlugin write\_prometheus

There might be more enabled plugins, it is fine. But they will not be used by the monitor at the moment.

4.Update plugin configurations as described below.

For "disk" plugin:

<Plugin disk>

Disk "/^[hs]d[a-f][0-9]?$/"

IgnoreSelected false

</Plugin>

For "write\_prometheus" plugin:

<Plugin write\_prometheus>

Port "9103"

</Plugin>

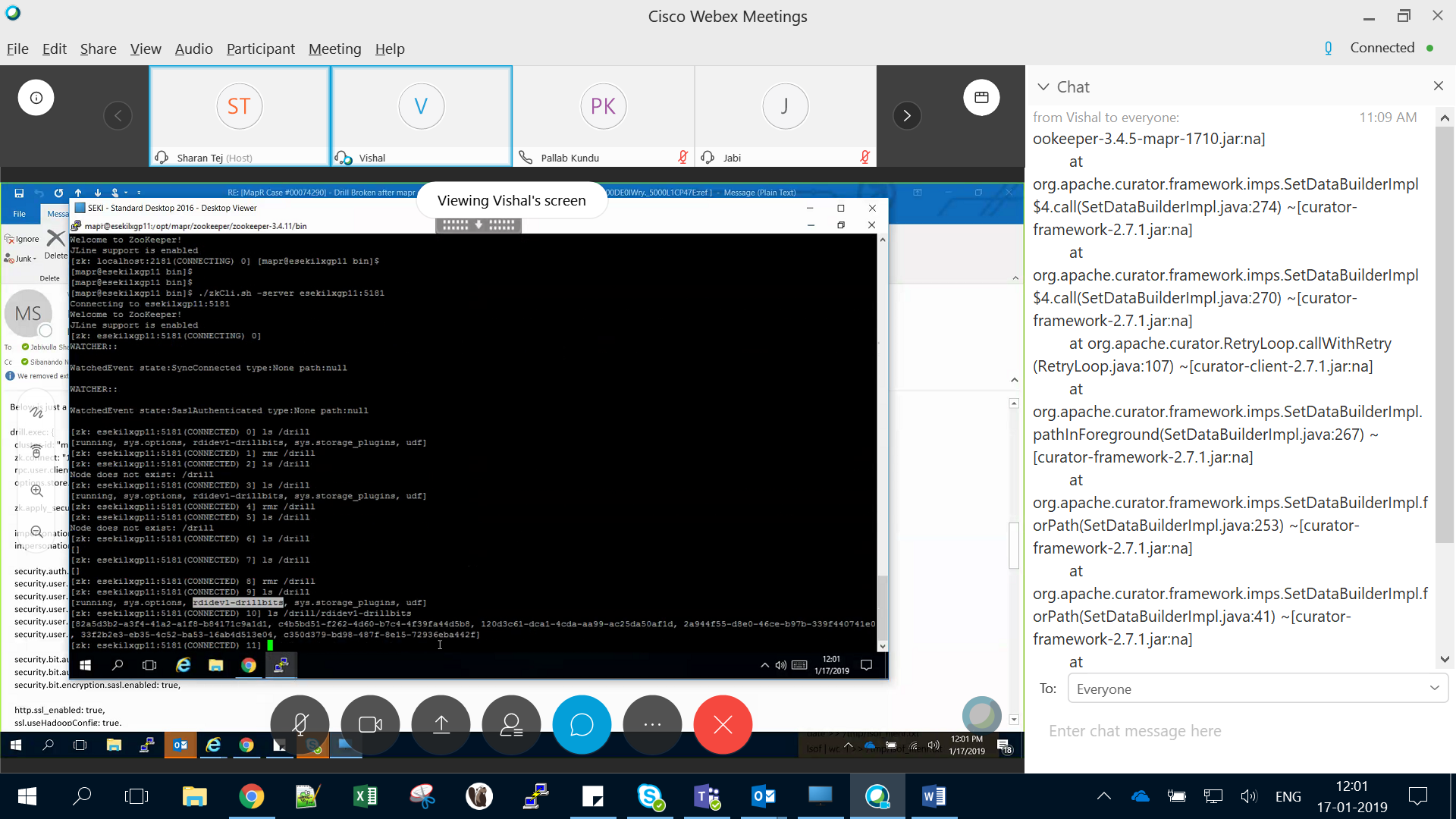
5.Restart collectd

systemctl restart collectd

6.Make sure that prometheus stats are available:

curl localhost:9103

**Drillbit**



1. **[root@esekilxgp09 conf]# cat drill-override.conf**

drill.exec: {

  cluster-id: "rdidev1-drillbits",

  zk.connect: "esekilxgp11.rnd.ki.sw.ericsson.se:5181,esekilxgp12.rnd.ki.sw.ericsson.se:5181,esekilxgp01.rnd.ki.sw.ericsson.se:5181",

  rpc.user.client.threads: "4",

  options.store.parquet.block-size: "268435456",

  zk.apply\_secure\_acl: true,

  impersonation.enabled: true,

  impersonation.max\_chained\_user\_hops: 3,

  security.auth.mechanisms: ["MAPRSASL", "PLAIN"],

  security.user.auth.enabled: true,

  security.user.auth.packages += "org.apache.drill.exec.rpc.user.security",

  security.user.auth.impl: "pam4j",

  security.user.auth.pam\_profiles: ["sudo", "login"],

  security.user.encryption.sasl.enabled: true,

  security.bit.auth.enabled: true,

  security.bit.auth.mechanism: "MAPRSASL",

  security.bit.encryption.sasl.enabled: true,

  http.ssl\_enabled: true,

  ssl.useHadoopConfig: true,

}

1. **[root@esekilxgp09 conf]# cat distrib-env.sh**

# This file is empty by default. Default Drill environment settings appear

# in drill-config.sh. Distributions can replace this file with a

# distribution-specific version that sets environment variables and options

# specific to that distribution. Users should not put anything in this file;

# put user options in drill-env.sh instead.

# MapR-specific environment settings for Drill

export HADOOP\_VERSION=`cat /opt/mapr/hadoop/hadoopversion`

export HADOOP\_HOME=${HADOOP\_HOME:-"/opt/mapr/hadoop/hadoop-${HADOOP\_VERSION}"}

#Enable JMX for MaprMonitoring

DRILL\_JMX\_OPTS="-Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=true -Dcom.sun.management.jmxremote.password.file=/opt/mapr/conf/jmxremote.password -Dcom.sun.management.jmxremote.access.file=/opt/mapr/conf/jmxremote.access -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.port=6090"

export DRILL\_JAVA\_OPTS="${DRILL\_JAVA\_OPTS} ${DRILL\_JMX\_OPTS} -Djava.io.tmpdir=/tmp/drill -Djava.security.auth.login.config=/opt/mapr/conf/mapr.login.conf -Dhadoop.login=hybrid\_keytab -Dzookeeper.sasl.client=true"

export DRILL\_LOG\_DIR=${DRILL\_LOG\_DIR:-"/opt/mapr/drill/drill-1.14.0/logs"}

export DRILL\_PID\_DIR=${DRILL\_PID\_DIR:-"/opt/mapr/pid"}

export MAPR\_IMPERSONATION\_ENABLED=${MAPR\_IMPERSONATION\_ENABLED:-"true"}

# Only set MAPR\_TICKETFILE\_LOCATION when invoked in context of drillbit setup NOT sqlline. It is expected

# to generate a separate ticket when sqlline is used.

if [ "$DRILLBIT\_CONTEXT" = "1" ]; then

    export MAPR\_TICKETFILE\_LOCATION=${MAPR\_TICKETFILE\_LOCATION:-"/opt/mapr/conf/mapruserticket"}

fi

export SQLLINE\_JAVA\_OPTS="${SQLLINE\_JAVA\_OPTS} -Ddrill.customAuthFactories=org.apache.drill.exec.rpc.security.maprsasl.MapRSaslFactory -Dzookeeper.sasl.client=true -Djava.security.auth.login.config=/opt/mapr/conf/mapr.login.conf"

1. **[root@esekilxgp09 conf]# cat drill-env.sh**

#

# Licensed to the Apache Software Foundation (ASF) under one

# or more contributor license agreements.  See the NOTICE file

# distributed with this work for additional information

# regarding copyright ownership.  The ASF licenses this file

# to you under the Apache License, Version 2.0 (the

# "License"); you may not use this file except in compliance

# with the License.  You may obtain a copy of the License at

#

# <http://www.apache.org/licenses/LICENSE-2.0>

#

# Unless required by applicable law or agreed to in writing, software

# distributed under the License is distributed on an "AS IS" BASIS,

# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

# See the License for the specific language governing permissions and

# limitations under the License.

#

# This file provides a variety of site-specific settings to control Drill

# launch settings. These are settings required when launching the Drillbit

# or sqlline processes using Java. Some settings are for both, some for one

# or the other.

#

# Variables may be set in one of four places:

#

#   Environment (per run)

#   drill-env.sh (this file, per site)

#   distrib-env.sh (per distribution)

#   drill-config.sh (Drill defaults)

#

# Properties "inherit" from items lower on the list, and may be "overridden" by items

# higher on the list. In the environment, just set the variable:

#

#   export FOO=value

#

# To support inheritance from the environment, you must set values as shown below:

#

#   export FOO=${FOO:-"value"}

#

# or a more specialized form.

# Amount of total memory for the Drillbit process. This value is defined as the limit

# that the startup script will try to enforce on the Drill JVM. The values can be

# defined in terms of percentage of the available system memory, or in terms of actual

# values, similar to how we define the actual JVM memory parameters like Heap Size.

# There is no default and depends on how much can be allotted on a machine.

# This enables Drill's memory auto-configuration logic to kick in, and should be unset

# if the intent is to not use the auto-configuration.

#export DRILLBIT\_MAX\_PROC\_MEM=${DRILLBIT\_MAX\_PROC\_MEM:-"13G"}

# Amount of heap memory for the Drillbit process. Values are those supported by

# the Java -Xms option. The default is 4G.

#export DRILL\_HEAP=${DRILL\_HEAP:-"4G"}

# Maximum amount of direct memory to allocate to the Drillbit in the format

# supported by -XX:MaxDirectMemorySize. Default is 8G.

#export DRILL\_MAX\_DIRECT\_MEMORY=${DRILL\_MAX\_DIRECT\_MEMORY:-"8G"}

# Native library path passed to Java. Note: use this form instead

# of the old form of DRILLBIT\_JAVA\_OPTS="-Djava.library.path=<dir>"

# The old form is not compatible with Drill-on-YARN.

# export DRILL\_JAVA\_LIB\_PATH="<lib1>:<lib2>"

# Value for the code cache size for the Drillbit. Because the Drillbit generates

# code, it benefits from a large cache. Default is 1G.

#export DRILLBIT\_CODE\_CACHE\_SIZE=${DRILLBIT\_CODE\_CACHE\_SIZE:-"1G"}

# Provide a customized host name for when the default mechanism is not accurate

#export DRILL\_HOST\_NAME=`hostname`

# Base name for Drill log files. Files are named ${DRILL\_LOG\_NAME}.out, etc.

# DRILL\_LOG\_NAME="drillbit"

# Location to place Drill logs. Set to $DRILL\_HOME/log by default.

#export DRILL\_LOG\_DIR=${DRILL\_LOG\_DIR:-$DRILL\_HOME/log}

# Location to place the Drillbit pid file when running as a daemon using

# drillbit.sh start.

# Set to $DRILL\_HOME by default.

#export DRILL\_PID\_DIR=${DRILL\_PID\_DIR:-$DRILL\_HOME}

# Default (Standard) CGroup Location: /sys/fs/cgroup

# Specify the cgroup location if it is different from the default

#export SYS\_CGROUP\_DIR=${SYS\_CGROUP\_DIR:-"/sys/fs/cgroup"}

# CGroup to which the Drillbit belongs when running as a daemon using drillbit.sh start .

# Drill will use CGroup for CPU enforcement only.

# Unset $DRILLBIT\_CGROUP by default

#export DRILLBIT\_CGROUP=${DRILLBIT\_CGROUP:-"drillcpu"}

# Custom JVM arguments to pass to the both the Drillbit and sqlline. Typically

# used to override system properties as shown below. Empty by default.

#export DRILL\_JAVA\_OPTS="$DRILL\_JAVA\_OPTS -Dproperty=value"

# As above, but only for the Drillbit. Empty by default.

#export DRILLBIT\_JAVA\_OPTS="$DRILLBIT\_JAVA\_OPTS -Dproperty=value"

# Process priority (niceness) for the Drillbit when running as a daemon.

# Defaults to 0.

#export DRILL\_NICENESS=${DRILL\_NICENESS:-0}

# Custom class path for Drill. In general, you should put your custom libraries into

# your site directory's jars subfolder ($DRILL\_HOME/conf/jars by default, but can be

# customized with DRILL\_CONF\_DIR or the --config argument. But, if you must reference

# jar files in other locations, you can add them here. These jars are added to the

# Drill classpath after all Drill-provided jars. Empty by default.

# custom="/your/path/here:/your/second/path"

# if [ -z "$DRILL\_CLASSPATH" ]; then

#   export DRILL\_CLASSPATH=${DRILL\_CLASSPATH:$custom}

# else

#   export DRILL\_CLASSPATH="$custom"

# fi

# Extension classpath for things like HADOOP, HBase and so on. Set as above.

# EXTN\_CLASSPATH=...

# Note that one environment variable can't be set here: DRILL\_CONF\_DIR.

# That variable tells Drill the location of this file, so this file can't

# set it. Instead, you can set it in the environment, or using the

# --config option of drillbit.sh or sqlline.

#-----------------------------------------------------------------------------

# The following are "advanced" options seldom used except when diagnosing

# complex issues.

#

# The prefix class path appears before any Drill-provided classpath entries.

# Use it to override Drill jars with specialized versions.

#export DRILL\_CLASSPATH\_PREFIX=...

# Enable garbage collection logging in the Drillbit. Logging goes to

# $DRILL\_LOG\_DIR/drillbit.gc. A value of 1 enables logging, all other values

# (including the default unset value) disables logging.

#export SERVER\_LOG\_GC=${SERVER\_LOG\_GC:-1}

# JVM options when running the sqlline Drill client.

# These are used ONLY in non-embedded mode; these

# are client-only settings. (The Drillbit settings are used when Drill

# is embedded.)

#export SQLLINE\_JAVA\_OPTS=""

# Arguments passed to sqlline (the Drill shell) at all times: whether

# Drill is embedded in Sqlline or not.

#export DRILL\_SHELL\_JAVA\_OPTS="..."

# Location Drill should use for temporary files, such as downloaded dynamic UDFs jars.

# Set to "/tmp" by default.

#

# export DRILL\_TMP\_DIR="..."

# Block to put environment variable known to both Sqlline and Drillbit, but needs to be

# differently set for both. OR set for one and unset for other.

#

# if [ "$DRILLBIT\_CONTEXT" = "1" ]; then

#   Set environment variable value to be consumed by Drillbit

# else

#   Set environment variable value to be consumed by Sqlline

# fi

#

export MAPR\_IMPERSONATION\_ENABLED=true

export MAPR\_TICKETFILE\_LOCATION=/opt/mapr/conf/mapruserticket

export DRILLBIT\_JAVA\_OPTS="-Djava.library.path=/opt/JPam-1.1/"

export DRILL\_JAVA\_OPTS="$DRILL\_JAVA\_OPTS -Djava.security.auth.login.config=/opt/mapr/conf/mapr.login.conf -Dzookeeper.sasl.client=true"

export DRILL\_JAVA\_OPTS="$DRILL\_JAVA\_OPTS -Dmapr\_sec\_enabled=true -Dhadoop.login=maprsasl\_keytab -Dzookeeper.saslprovider=com.mapr.security.maprsasl.MaprSaslProvider -Dmapr.library.flatclass"

[root@esekilxgp09 conf]#

**Description**

Initially, issue was with the properties defined in drill-override.conf file.

After the file is modified as suggested, new property was added which enabled secure acl's on drill znodes.

Due to which there was permission issue in creating a znode under /drill/sys.options because the initial znodes were created without any acl's.

We have set zk.apply\_secure\_acl: true in drill-override.conf and zookeeper.sasl.client=true in drill-env.sh and distrib-env.sh, deleted the /drill znode and restarted the drillbits.

Now the znodes got created with proper acl's and the issue got resolved.

@Vishal Regarding your question about the zk acl property :

- without zk.apply\_secure\_acl, znodes are created with below acl.

[zk: sharan61:5181(CONNECTED) 4] getAcl /drill/demo.mapr.com-drillbits 'world,'anyone

: cdrwa

[zk: sharan61:5181(CONNECTED) 5] getAcl /drill/demo.mapr.com-drillbits/bead46af-41bd-40d3-8fe8-d97f6d1ee5a8

'world,'anyone

: cdrwa

Here, everyone was write access on the znode.

- with zk.apply\_secure\_acl, znodes are created with below acl.

[zk: sharan02.tej.com:5181(CONNECTED) 2] getAcl /drill/me.sharantej.com-drillbits 'world,'anyone

: r

'sasl,'mapr

: cdrwa

[zk: sharan02.tej.com:5181(CONNECTED) 3] getAcl /drill/sys.options

'sasl,'mapr

: cdrwa

Here, only the super user(mapr) has the write privilige.

That is the difference.

**Livy issue :**

It seems a step in upgrade process is missed. Please see below document.

<https://mapr.com/docs/61/UpgradeGuide/RestartingClusterServices.html#RestartingClusterServices>

It seems step 2 to 6 are missed or may be other also. Can you please review that you have performed all steps here.

Step 2 talks about removing /opt/mapr/conf/mapruserticket . In 6.0.1 impersonation is set to false for this ticket and in 6.1 it needs to be true. Once you perform above steps, new ticket should get created with impersonation true.

I was able to reproduce this issue on my internal environment while upgrading from 6.0.1 to 6.1.

This will fix issue. I tested it my environment.

Finally all issues related to Spark jobs are fixed in DEV cluster. During the investigation, we come across some other unexpected issues as well. To add on top of Vishal’s already mentioned points

**Problem Statement:**

1. Spark jobs fails which are submitted using livy from gp02
2. In DEV posix node all maprfs directories/files ownership changes to mapr

Both the case is tracked from JIRA ticket

<https://wcdma-jira.rnd.ki.sw.ericsson.se/browse/RDIOP-222>

<https://wcdma-jira.rnd.ki.sw.ericsson.se/browse/RDIOP-230>

**Intermediate Solution:**

1. export KRB5CCNAME property addition mentioned by Vishal already
2. mfs -setace permission change mentioned by Vishal already

**Permanent Solution:**

1. Regenerate of mapr service ticket with impersonation in POSIX client node (mentioned by Vishal already). We need to make sure that maprservice ticket has impersonation “true”.
2. Follow the post upgrade steps of MapR core (Ref - <https://mapr.com/docs/61/UpgradeGuide/RestartingClusterServices.html>). Because at MapR 6.1 /opt/mapr/conf/mapruserticket ticket require impersonation “true”.
3. At DEV cluster, we did following at esekilxgp08
4. systemctl stop mapr-warden
5. rm /opt/mapr/conf/mapruserticket
6. /opt/mapr/server/configure.sh -R
7. systemctl start mapr-warden
8. Copied the newly generated mapruserticket to all other nodes except esekilxgp02
9. Make sure the file owner is mapr:maprg

--   
**Hive 2.3 and spark 2.3.1**

**Parameters need to be added: in hive-site.xml**

<property>

    <name>hive.metastore.sasl.enabled</name>

    <value>true</value>

</property>

<property>

    <name>hive.server2.authentication</name>

    <value>PAM</value>

</property>

for Hive 2.3 and spark 2.3.1

**Pem generate command**

**/opt/mapr/server/manageSSLKeys.sh convert -N rdidev1 ssl\_truststore ssl\_truststore.pem**

**Scandry index/Secondary index**

<https://mapr.com/support/s/article/Secondary-index-creates-operation-failed-with-error-Cluster-Gateways-are-not-configured?language=en_US>

# Secondary index creates operation failed with error "Cluster Gateways are not configured"

Secondary indexes can be created only on MapR-DB JSON tables to provide efficient access to a wider range of queries on data in MapR-DB. Secondary index creates operation failing for manually installed cluster.

Jan 10, 2018•Generic - Issue Resolution

**Environment**

E.g.: MapR 4.0.1, Hbase 0.98, Redhat 5.5 etc

MapR v6.0 and above

**Symptom**

1. Below error messages are shown in </var/log/messages>: <Error Messages if appropriate> 2. MCS shows alerts "xxx is down" 3. Spark jobs are hung.

The general sympton is creating SI for a table will fail with error "Cluster Gateways are not configured"

maprcli table index add -path /apps/my\_users -index newIndex -indexedfields name

ERROR (22) -  Failed to add index for table: /apps/my\_users : Cluster Gateways are not configured

2017-12-19 21:38:34,0634 ERROR Client fs/client/fileclient/cc/[dbclient.cc](http://dbclient.cc/):686 Thread: 24742 AddTableIndex failed,

**Solution**

**Root Cause**

Examples: 1. Jobtracker configuration is incorrect in /opt/mapr/hue/hue-<version>/desktop/conf/hue.ini 2. No active Jobtracker for the cluster.

When you create secondary index on a table this internally requires replication gateway to replicated data between the table and the secondary index. If the installation of cluster done manually we often do not install gateway on local cluster which could lead this issue.

**Solutions**

<Clarify whether this is a work-around or a solution that may not apply. Where possible, provide for the customer a way to cross-check whether they implemented your solution correctly>. Code format in Courier New 1. Code xxx 2. Code yyy 3. Code zzz

You need to install the replication gateways. Since the source JSON table and the secondary index are on the same volume within a cluster, configure an [intra-cluster gateway](https://maprdocs.mapr.com/home/Gateways/Gateways-replication-Tables.html#GatewaysforTableReplicati_29656856-d3e64__section_dzb_kg1_z1b).

$ yum install mapr-gateway

$ /opt/mapr/server/[configure.sh](http://configure.sh/) -R

$  maprcli cluster gateway set -dstcluster <local\_cluster> -gateways <gateway\_hostname>

Or

maprcli cluster gateway set -dstcluster rdidev1 -gateways exekilxgp07.rnd.ki.sw.ericsson.se

**creating Index**

maprcli table index add -path /mapr/rdidev1/user/e/metadata3 -index rnsTime -indexedfields Subnetwork:ASC,Date:ASC

maprcli table index add -path /mapr/rdidev1/user/mapr/test022 -index ind01 -indexedfields name:ASC,Date:ASC

maprcli cluster gateway set -dstcluster rdidev1 -gateways esekilxgp07.rnd.ki.sw.ericsson

**Spark Cache**

Spark not using local discs for cache on all datanodes on rdiprod1

We have noticed significantly worse performance on the mapr cluster for some of our larger spark jobs compared to the cloudera cluster used by other teams, we are not expecting this to be the case since this cluster has better capabilities.

Upon investigation, we have noticed that datanodes that have been added later than the original nodes for rdiprod1 have not been set up so that spark uses local discs for caching, but instead they use mapr root volume.

The task is to set upp all datanodes such that spark is using local discs for caching.

**Solution**

Spark Volume

1. Shutdown NM on the faulty node.

2. remove from spark volume from MCS.

3. Clean up the directory structure for spark under local volumes for the node.

4. maprcli volume create -name mapr.$(hostname -f).local.spark -path /var/mapr/local/$(hostname -f)/spark -replication 1 -localvolumehost $(hostname -f)

5. Check the volume for mount and size from MCS.

6. Restart NM

@Anju:- please follow the steps and implement node by node. Once done on all nodes we need to restart RM in a failover mode. Please bear in mind it’s a 0 down time ticket.

maprcli volume create -name mapr.$(hostname -f).local.spark -path /var/mapr/local/$(hostname -f)/spark -replication 1 -localvolumehost $(hostname -f)

hadoop fs -ls /var/mapr/local/esekilx5539.rnd.ki.sw.ericsson.se/spark

hadoop fs -rmr /var/mapr/local/esekilx5539.rnd.ki.sw.ericsson.se/spark

**Reading Cores**

gdb -ex 'set confirm off' -ex 'set pagination off' -ex 'thread apply all bt' -ex 'quit' -c <core\_file>  /opt/mapr/server/mfs  >  /tmp/gdb\_`hostname`.out

**Queue setup in Fair Scheduler**

**Jira Ticket** - <https://wcdma-jira.rnd.ki.sw.ericsson.se/browse/RDIOP-296>

**Changes in DEV Cluster:**

Added the following block in file

/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/fair-scheduler.xml

<queue name="dataexport">

  <maxResources>479423 mb,50 vcores,0 disks</maxResources>

</queue>

Affected Nodes: All DEV cluster nodes (esekilxgp[01-02,07-12].rnd.ki.sw.ericsson.se)

**Changes in PROD Cluster:**

Added the following block in file

/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/fair-scheduler.xml

<queue name="dataexport">

  <maxResources> 1024000 mb,150 vcores,0 disks</maxResources>

</queue>

Affected Nodes: All PROD cluster nodes (esekilx[5517-5520,5530-5533,5537-5540,5634-5638,5640-5647].rnd.ki.sw.ericsson.se)

**Test Steps:**

The queue changes can be seen from RM’s web interface, at https://\*ResourceManager URL\*/cluster/scheduler

Example Spark Job –

/opt/mapr/spark/spark-2.3.1/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --queue dataexport /opt/mapr/spark/spark-2.3.1/examples/jars/spark-examples\_2.11-2.3.1-mapr-1808.jar 4

**YARN performance improvement**

**We have added below parameters in yarn-site.xml for YARN performance improvement.**

<property>

<name>yarn.scheduler.minimum-allocation-mb</name>

<value>2048</value>

<description>Every contianer will get a minimum of 2GB memory</description>

</property>

<property>

<name>yarn.scheduler.maximum-allocation-mb</name>

<value>24576</value>

<description>Every contianer can get a maximum of 24GB memory to ensure at max 3 containers can run on 1 node</description>

</property>

<property>

<name>yarn.scheduler.minimum-allocation-vcores</name>

<value>1</value>

<description>Every contianer will get a minimum of 1 vcore</description>

</property>

<property>

<name>yarn.scheduler.maximum-allocation-vcores</name>

<value>8</value>

<description>Every contianer can get a maximum of 8 vcores to ensure at max 2 containers using 8 vcores each can run on 1 node</description>

</property>

<property>

<name>yarn.nodemanager.pmem-check-enabled</name>

<value>false</value>

<description>Container will be killed if it uses more than allocated physical memory</description>

</property>

<property>

<name>yarn.nodemanager.vmem-check-enabled</name>

<value>false</value>

<description>Container will be killed if it uses more than allocated virtual memory based on vmem-pmem ratio.Currently disabled</description>

</property>

<property>

<name>yarn.nodemanager.vmem-pmem-ratio</name>

<value>2.1</value>

<description>Ratio of Virtual memory to be allocated to container wrt Physical memory</description>

</property>

<property>

<name>yarn.nodemanager.resource.memory-mb</name>

<value>49152</value>

<description>Total memory in MB at max provided for Nodemanager to assign to containers. Assigned it to 50 percent with the assumption that no custom scripts are executed by developers on cluster nodes.</description>

</property>

<property>

<name>yarn.nodemanager.resource.cpu-vcores</name>

<value>16</value>

<description>Total vcores at max provided for Nodemanager to assign to containers. Assigned it to 50 percent with the assumption that no custom scripts are executed by developers on cluster nodes.</description>

</property>

**Enable Spark Dynamic Allocation**

**We have added below parameters in spark-default.conf** 

#Dynamic Allocation

spark.dynamicAllocation.enabled         true

spark.shuffle.service.enabled           true

spark.dynamicAllocation.minExecutors    0

spark.dynamicAllocation.maxExecutors    50

spark.executor.instances                2

**We have added below parameters in yarn-site.xml**

<property>

   <name>yarn.nodemanager.aux-services</name>

   <value>mapreduce\_shuffle,mapr\_direct\_shuffle,spark\_shuffle</value>

</property>

<property>

   <name>yarn.nodemanager.aux-services.spark\_shuffle.class</name>

   <value>org.apache.spark.network.yarn.YarnShuffleService</value>

</property>

**Copy the file to yarn lib**

cp -p /opt/mapr/spark/spark-2.2.1/yarn/spark-2.2.1-mapr-1901-r2-yarn-shuffle.jar /opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/yarn/lib

**HUE metadata migration to MySQL** 

1. **yum install MySQL-python.x86\_64 mysql-connector-python.noarch**
2. **Install pip**   
      
   curl [https://bootstrap.pypa.io/get-pip.py -o get-pip.py](https://bootstrap.pypa.io/get-pip.py%20-o%20get-pip.py)python get-pip.py

1. **Install mysql-python moudule**

pip install --upgrade pip   
pip install ConfigParser MySQL MySQL-python mysqlclient --no-cache-dir

1. **Add below parameters in hue.ini**

engine=mysql

 host=hadoop-c02n09.ss.sw.ericsson.se

 port=3306

 user=hue

 password=Hue@12345

 name=hue

1. **Perform the initial data migration:**

cd /opt/mapr/hue/hue-<version>

source ./build/env/bin/activate

hue syncdb --noinput

hue migrate

deactivate

Local MapR Repository Creation 

1. **Download MapR Packages**   
      
    url : http://stage.mapr.com/ericsson/

User id : ericsson

pass : \*\*\*\*\*\*\*\*\*\*   
   
wget/download the MapR core and MEP packages from above mentioned URL.

CORE: mapr-v6.1.0GA.rpm.tgz   
MEP: mapr-mep-v6.0.0.201810030946.rpm.tgz 

1. **Untar the two packages to get all the RPMs.**

Copy the two packages to /var/www/html/yum/base   
tar -xzf mapr-v6.1.0GA.rpm.tgz   
tar -xzf mapr-mep-v6.0.0.201810030946.rpm.tgz

1. **Create softlink**

Create softlink under /var/www/html/yum/base directory   
ln -s /var/www/html/yum/base/v6.1.0 mapr\_core   
ln -s /var/www/html/yum/base/MEP/MEP-6.0 mapr\_eco

1. **Create repository**   
      
   createrepo /var/www/html/yum/mapr\_eco   
   createrepo /var/www/html/yum/mapr\_core

1. **Install and start httpd(apache) service**   
   yum install httpd   
   systemctl start httpd

1. **Crete repo file under /etc/yum.repos.d**   
      
   [mapr\_core]

name=MapR Technologies, Inc.

baseurl=http://10.142.0.5/yum/mapr\_core/

enabled=1

gpgcheck=0

proxy=\_none\_

[mapr\_ecosystem]

name=MapR Technologies, Inc.

baseurl=http://10.142.0.5/yum/mapr\_eco/

enabled=1

gpgcheck=0

proxy=\_none\_ 

**Drill Memory**

**I have modified the below memory parameters in warden.drillbit.conf**

DRILL\_MAX\_DIRECT\_MEMORY:-"16G"

service.env=DRILLBIT\_MAX\_PROC\_MEM=25G

service.heapsize.min=20480

service.heapsize.max=25600

**And below parameters have been updated in /opt/mapr/drill/drill-1.13.0/conf/drill-env.sh**

export DRILL\_MAX\_DIRECT\_MEMORY=${DRILL\_MAX\_DIRECT\_MEMORY:-"16G"}

export DRILL\_HEAP=${DRILL\_HEAP:-"8G"}

export DRILLBIT\_CODE\_CACHE\_SIZE=${DRILLBIT\_CODE\_CACHE\_SIZE:-"1G"}

**HIVE-TEZ**

**We have added below parameters in hive-site.xml for HIVE performance improvement**

<property>

<name>hive.tez.container.size</name>

<value>24576</value>

<description> TEZ application master size has been set to 24GB, so that at least 2 TEZ contains can run per node </description>

</property>

<property>

<name>hive.tez.java.opts</name>

<value>-Xmx18432m</value>

</description> HEAP seize for TEZ application master </description>

</property>

Streamsets Installation and configuration

1. **Download Streamset Datacollector full RPM from**[**https://streamsets.com/opensource**](https://streamsets.com/opensource).

wget <https://s3-us-west-2.amazonaws.com/archives.streamsets.com/datacollector/3.6.1/rpm/el7/streamsets-datacollector-3.6.1-el7-all-rpms.tar>

1. **Untar all the RPMs**   
      
   tar -xf streamsets-datacollector-3.6.0-el7-all-rpms.tar
2. **Install RPMs and Install JAVA**

For our solution we need below RPMs   
streamsets-datacollector-3.6.0-1.noarch.rpm   
streamsets-datacollector-hdp\_2\_6-hive2-lib-3.6.0-1.noarch.rpm   
streamsets-datacollector-mapr\_6\_0-mep5-lib-3.6.0-1.noarch.rpm   
streamsets-datacollector-mapr\_6\_0-lib-3.6.0-1.noarch.rpm   
streamsets-datacollector-mysql-binlog-lib-3.6.0-1.noarch.rpm   
   
**Download Java**

wget --no-check-certificate --no-cookies --header "Cookie: oraclelicense=accept-securebackup" https://download.oracle.com/otn-pub/java/jdk/8u191-b12/2787e4a523244c269598db4e85c51e0c/jdk-8u191-linux-x64.rpm

1. **Configure Streamset to run as mapr user**   
      
   By default streamsets runs unser sdc user, which will be created automaticaaly during rpm installtation. We need to change this, otherwise streamset will not be able to write to MapR.   
      
   a. update User & Group as mapr in /usr/lib/systemd/system/sdc.service file    
   b. chown -R mapr:mapr /var/log/sdc

c. chown -R mapr:mapr /var/lib/sdc   
d. mkdir -p /opt/streamsets-datacollector/streamsets-libs-extras/   
e. mkdir -p /opt/streamsets-datacollector/streamsets-libs-extras/

f. chown -R mapr:mapr /etc/sdc

1. **Install MapR-client 6.0.0**   
      
   export SDC\_HOME=/opt/streamsets-datacollector

export SDC\_CONF=/etc/sdc

export MAPR\_MEP\_VERSION=5   
   
/opt/streamsets-datacollector/bin/streamsets setup-mapr   
Please enter the MapR version (default 6.0.0): 6.0.0   
Please enter the absolute path of MapR Home (default /opt/mapr): /opt/mapr

1. **Upload mysql java connector and mysql binlog connector.**
2. **Change mysql binlog property in /etc/my.cnf**   
   mysqld]

server-id=1

log-bin=mysql-bin

binlog\_format = Row

1. **Open streamset from http://ip\_address:18630**

Monitoring Using Telegraf & InfluxDB

**Downdoal the required package:**

wget https://dl.influxdata.com/telegraf/releases/telegraf-1.7.1-1.x86\_64.rpm

wget <https://dl.influxdata.com/influxdb/releases/influxdb-1.6.0.x86_64.rpm>

**Install the package:** 

yum install telegraf-1.7.1-1.x86\_64.rpm influxdb-1.6.0.x86\_64.rpm -y

**Generate telegraf.conf file:**

telegraf --output-filter influxdb config >/etc/telegraf/telegraf.conf

**Edit the telegraf.conf file for influxdb output plugin**

[[outputs.influxdb]]

  urls = ["http://esekilxgp02.rnd.ki.sw.ericsson.se:8086"]

  database = "telegraf\_metrics"

  retention\_policy = ""

  write\_consistency = "any"

  timeout = "5s"

**Don’t change other parameter and hashout any other parameter, if enabled.**

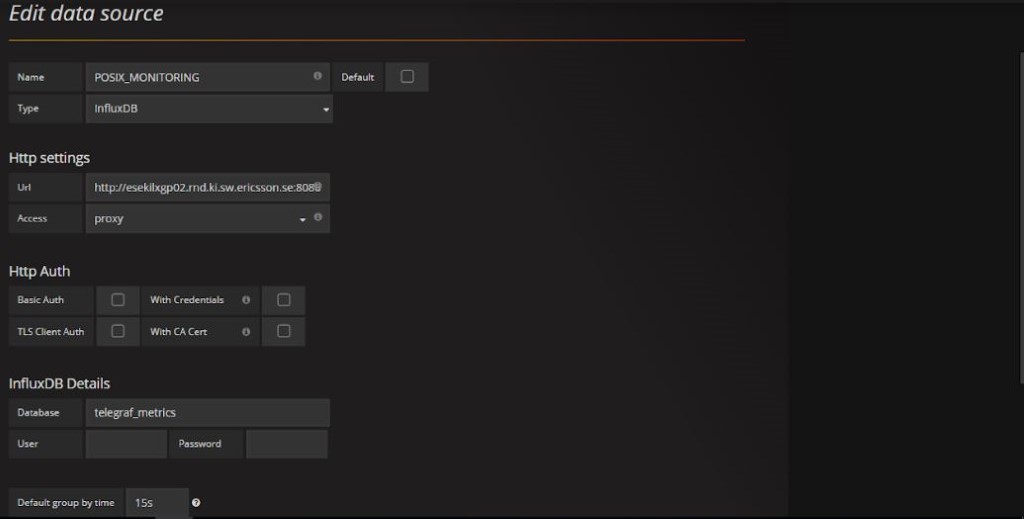
**Start services for telegraf and Influxdb:**

systemctl start telegraf

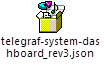
systemctl start influxdb

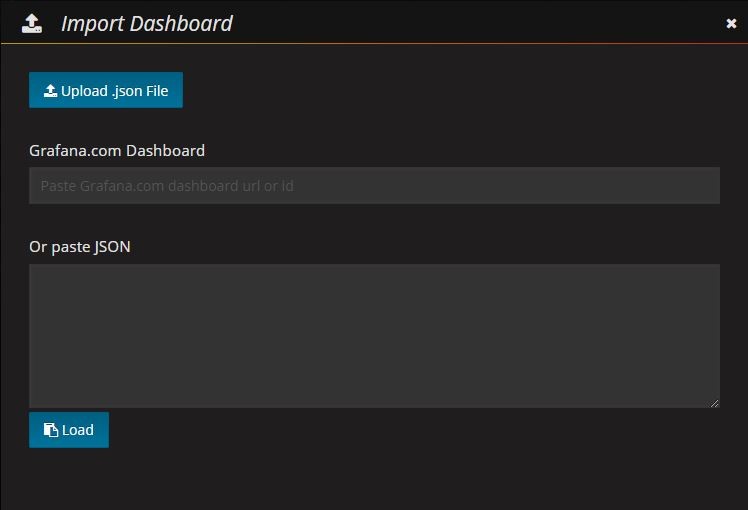
**Add DataSource to Grafana:**

The url for influxDB will be : [http://esekilxgp02.rnd.ki.sw.ericsson.se:8086](http://esekilxgp02.rnd.ki.sw.ericsson.se:8086/)Database name will be: telegraf\_metrics (Same as in telegraf.conf)

****

**Import DashBoard to Grafana for this Monitoring:**

****

****

MARIADB HIGH AVAILIBILITY

**In Server1 do the following steps**

1. **Edit /etc/my.cnf in server1**:

[mysqld]

server-id=1

log-bin=mysql-bin

binlog\_format = MIXED

1. **Restart the database**   
   systemctl stop mariadb   
   systemctl start mariadb
2. **User creation for replication**   
   create user 'mapr'@'server1' identified by 'Had00p!';   
   create user 'mapr'@'server2' identified by 'Had00p!';
3. **Grant access to the databases**   
   grant all privileges on \*.\* to 'mapr'@'server1';   
   grant all privileges on \*.\* to 'mapr'@'server2';   
   grant replication slave on \*.\* to 'mapr'@server2' identified by 'Had00p!';   
   flush privileges;
4. **Check the binlog\_format:**

MariaDB [(none)]> show variables like 'binlog\_format';

+---------------+-------+

| Variable\_name | Value |

+---------------+-------+

| binlog\_format | MIXED |

+---------------+-------+

1. **Flush the tables for read only access**.   
   flush tables with read lock;
2. **Check current log status**   
   MariaDB [(none)]> show master status\G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

File: mysql-bin.000002

Position: 1474

Binlog\_Do\_DB:

Binlog\_Ignore\_DB:

1 row in set (0.00 sec) 

1. **Backup all the existing databases**   
   mysqldump -u mapr -p --database mysql > /home/mapr/mysql.mysql   
   mysqldump -u mapr -p --database hive > /home/mapr/hive.mysql   
   mysqldump -u mapr -p --database hue > /home/mapr/hue.mysql   
   mysqldump -u mapr -p --database oozie > /home/mapr/oozie.mysql
2. **scp all these backup file to server2**.   
   scp -p /home/mapr/\*.mysql mapr@server2:/home/mapr/

**In Server2 do the following steps**

1. **Edit /etc/my.cnf in server1:**

[mysqld]

server-id=2

log-bin=mysql-bin

binlog\_format = MIXED

1. **Restart the database**   
   systemctl stop mariadb   
   systemctl start mariadb
2. **User creation for replication**   
   create user 'mapr'@'server1' identified by 'Had00p!';   
   create user 'mapr'@'server2' identified by 'Had00p!';
3. **Grant access to the databases**   
   grant all privileges on \*.\* to 'mapr'@'server1';   
   grant all privileges on \*.\* to 'mapr'@'server2';   
   grant replication slave on \*.\* to 'mapr'@server1' identified by 'Had00p!';   
   flush privileges;
4. **Check the binlog\_format:**

MariaDB [(none)]> show variables like 'binlog\_format';

+---------------+-------+

| Variable\_name | Value |

+---------------+-------+

| binlog\_format | MIXED |

+---------------+-------+

1. **Restore all the databases**.

mysql mysql -u mapr -p < mysql.mysql

mysql mysql -u hive -p < hive.mysql

mysql mysql -u hue -p < hue.mysql

mysql mysql -u oozie -p < oozie.mysql

1. **Flush the tables for read only access.**   
   flush tables with read lock;

1. **Check current log status**   
   MariaDB [(none)]> show master status\G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

            File: mysql-bin.000003

        Position: 518215

    Binlog\_Do\_DB:

Binlog\_Ignore\_DB:

1 row in set (0.00 sec)

1. **Sync server1 log**   
   change master to master\_host='server1', master\_user='mapr', master\_password='Had00p!', master\_log\_file='mysql-bin.000002', master\_log\_pos=1474;

1. **Convert the current instance to act as slave.**   
   MariaDB [(none)]> start slave;

Query OK, 0 rows affected (0.00 sec)

1. **Check the slave status for any error**   
   MariaDB [(none)]> show slave status\G
2. **Unlock all the tables:**   
   MariaDB [(none)]> unlock table;

Query OK, 0 rows affected (0.00 sec)   
   
   
**In Server1 do the following steps**

1. **Sync server1 log**   
   change master to master\_host='server2', master\_user='mapr', master\_password='Had00p!', master\_log\_file='mysql-bin.000003', master\_log\_pos=518215;

1. **Convert the current instance to act as slave.**   
   MariaDB [(none)]> start slave;

Query OK, 0 rows affected (0.00 sec)

1. **Check the slave status for any error**   
   MariaDB [(none)]> show slave status\G
2. **Unlock all the tables:**   
   MariaDB [(none)]> unlock table;

Query OK, 0 rows affected (0.00 sec)

***PostgreSQL Installation***

**Add Postgres Yum Repository :**

* Adding PostgreSQL yum repository in our Red hat Linux distribution
* # yum install https://download.postgresql.org/pub/repos/yum/10/redhat/rhel-7-x86\_64/pgdg-redhat10-10-2.noarch.rpm

**Install PostgreSQL Server :**

* After adding PostgreSQL yum repository in our Red hat Linux distribution, use the following command to install PostgreSQL server and client packages.

# yum install postgresql10 postgresql10-libs postgresql10-contrib postgresql10-server

**Initialize PostgreSQL Database :**

* Due to some policies for Red Hat based distributions, the PostgreSQL installation will not be active for automatic start or have the database initialized automatically. To complete your database installation, we need to initialize your database before using it for first time.

# /usr/pgsql-10/bin/postgresql-10-setup initdb

**Start and Enable PostgreSQL Server :**

* After database initialize completes, start PostgreSQL service and enable PostgreSQL service to auto start on system boot.

For RHEL 7

# systemctl start postgresql-10

# systemctl enable postgresql-10

# systemctl status postgresql-10

For RHEL 6

# service postgresql-10 start

# chkconfig postgresql-10 on

**Verify PostgreSQL Installation :**

* After completing above steps, you have installed PostgreSQL 10 on your server, Let’s log in to postfix to verify that installation completed successfully.

# su - postgres -c "psql"

psql (10.0)

Type "help" for help.

   postgres=#

* You may create password for user postgres for security purpose.

postgres=# \password postgres

***STREAMING REPLICATION (HIGH AVAILABILITY)***

**Master Server Configuration :-**

Begin with master server configuration updates that we need to make in-order to enable replication.

root@esekilx5916# vi /var/lib/pgsql/10/data/postgres.conf       /\*Edit configuration file\*/

Update the following parameters :

wal\_level = hot\_standby               /\* to enable streaming replication \*/

archive\_mode = on /\* to enable archive process \*/

archive\_command = 'cp %p /var/lib/pgsql/10/archive/%f'             /\* create myarchive folder and give postgres permission \*/

max\_wal\_senders = 10                                          /\* number of parallel wal senders to be initiated \*/

wal\_keep\_segments = 50                                       /\* maintining 50 xlog files in pg\_xlog directory \*/

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

root@esekilx5916#  /etc/init.d/postgresql-10 restart   /\* Restart database engine to get all the changes into effect in Master server \*/

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

root@esekilx5916# . /psql -d postgres                   /\* login to postgres primary server \*/

Next thing we need to do is take data snapshot of data from master and later move that to slave server

postgres# select pg\_start\_backup('streaming');    /\* we need to initiate the base backup \*/

postgres#\!

root@esekilx5916# tar cfP /tmp/db\_file\_backup.tar /var/lib/pgsql/10/data/

postgres# select pg\_stop\_backup();  /\* after taking base backup login back to primary server and stop the backup \*/

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

Now we need to grant access to read Master servers WAL logs from standby server.

Edit host base auth configuration file:

root@esekilx5916# vi /var/lib/pgsql/10/data/pg\_hba.conf

host    replication     postgres       134.138.193.181/32  trust     /\*  uncomment and add the standby host name or its IP address \*/

root@esekilx5916# scp  /tmp/db\_file\_backup.tar ezchast@esekilx5915:/var/lib/pgsql/10/data  /\* Copy data from Master to Standby\*/

root@esekilx5916# /etc/init.d/postgresql-10 restart   /\* Restart database engine to get all the changes into effect in Master server\*/

**Slave server configuration:**

If PostgreSQL is running Stop the Standby Server and make the changes

  --Unzip master server data snapshot file that is copied into this server

root@esekilx5915# tar xvfP /tmp/db\_file\_backup.tar

 --Remove postmaster.pid so standby server does not see the primary server’s pid as its own

root@esekilx5915# rm -rf  /var/lib/pgsql/10/data/postmaster.pid

  --Now edit configuration file and tweak hot\_standby variable.

root@esekilx5915# vi /var/lib/pgsql/10/data/postgresql.conf    /\* open standby server postgresql.conf file and make the changes \*/

hot\_standby = on    /\* change this parameter to make standby server as recovery server \*/

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

  --Now we need to create a recovery.conf file for this slave server to start receiving logs from master.

root@esekilx5915# create recovery.conf           /\* create recovery.conf file inside standby server and add these parameters\*/

  -- Edit this recovery.conf file and update standby server settings

root@esekilx5915# vi /var/lib/pgsql/10/data/recovery.conf

standby\_mode = 'on'/\* when standby parameter is enabled,the PostgreSQL server will work as standby. It will continuously wait for the XLOG records, Using restore\_command or Primary\_conninfo \*/

primary\_conninfo = 'host=esekilx5916.rnd.ki.sw.ericsson.se port=5432' /\*when  primary server connection information is set PostgreSQl will try to connect to master server using this connection string and receive XLOG records continuously\*/

trigger\_file = '/tmp/trigger.5432'                    /\* trigger.5432 file we need to create if primary goes down and standby server will come up\*/

restore\_command = 'cp /var/lib/pgsql/10/archive/%f %p'/\* we need to specify the archive path to restore the transaction from Master server \*/

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

root@esekilx5915# chown postgres.postgres /var/lib/pgsql/10/data/recovery.conf  /\* Update permissions on recovery.conf file \*/

--After making above changes in standby server we need to start the standby server--

root@esekilx5915# /etc/init.d/postgresql-10 start

root@esekilx5915# ./psql -d postgres

postgres# \dt/\* execute to list the tables and you should be seeing the tables which was in primary server \*/

Now we have successfully started our streaming replication hot-standby server. It may take few minutes for the server to be fully up and running as it needs to sync logs with master.

---To test replication, simple add/insert into a table on master server and query the same from slave server—

**RDI Dev Cluster Upgrade IM Plan.**

1. Create repos pointing to MapR 6.1 and MEP 6.0. Turn on the repos only the night of 9th.

2. Backup of all nodes must be taken three days prior to upgrade.

clush -b -g dev ‘cp /opt/mapr /root/backup’

3. Bring down the cluster on 10th morning

a. yarn application -list

b. yarn application kill

c. stop services from MCS (mapr-nfs,NM,RM)

d. Shut down all the POSIX nodes

e. Shut down the warden

f. Shut down the zookeepers

g. ps -ef | grep v1funcr | awk '{print $2}' | xargs kill -9

h. ps -ef | grep mapr | awk '{print $2}' | xargs kill -9

4. remove all the MAPR patches

a. clush -b -g dev ‘yum erase -y mapr-patch’

b. For edge node this need to be done by logging to individual nodes.

5. Update the cluster with MapR Core 6.1 and MEP 6.0

a. clush -b -g dev ‘ yum update -y mapr-\\*’

b. upgrade the POSIX the same way

c. /opt/mapr/server/configure.sh -R

6. Configure the Ecosystem components especially Spark, Livy, Hive, Impala, Hue, Oozie. Gp02 should be given special care for R libraries. We need to test Log Monitoring and Metrics Monitoring Modules. Let us copy the .xml of respective node to the newly created conf.

**RDI Production Cluster Upgrade IM Plan.**

1. Create repos pointing to MapR 6.1 and MEP 6.0. Turn on the repos only the night of 25th.

2. Backup of all nodes must be taken three days prior to upgrade.

clush -b -g prod “ mkdir /root/backup”

clush -b -g prod “ chmod 755 /root/backup”

clush -b -g prod "find /opt/mapr -type d -name conf\\* -exec cp --parent -r \{\} /root/backup \;"

clush -b -g prod "find /opt/mapr -type d -name hue\\* -exec cp --parent -r \{\} /root/backup \;"

clush -b -w esekilx5645.rnd.ki.sw.ericsson.se esekilx5645.rnd.ki.sw.ericsson.se "find /opt/mapr -type d -name elasticsearch\\* -exec cp --parent -r \{\} /root/backup \;"

clush -b -g prod "find /opt/mapr -type d -name fluentd\\* -exec cp --parent -r \{\} /root/backup \;"

clush -b -w esekilx5645.rnd.ki.sw.ericsson.se esekilx5646.rnd.ki.sw.ericsson.se "find /opt/mapr -type d -name kibana\\* -exec cp --parent -r \{\} /root/backup \;"

3. Bring down the cluster on 26th morning using mapr user.

a. yarn application -list

b. yarn application -kill <app ID>

c. stop services from MCS (mapr-nfs,NM,RM)

d. Shut down all the POSIX nodes

e. Shut down the warden

f. Shut down the zookeepers

g. ps -ef | grep v1funcr | awk '{print $2}' | xargs kill -9

h. ps -ef | grep mapr | awk '{print $2}' | xargs kill -9

4. remove all the MAPR patches

a. clush -b -g prod ‘yum erase -y mapr-patch’

b. For edge node this need to be done by logging to individual nodes.

5. Update the cluster with MapR Core 6.1 and MEP 6.0

a. clush -b -g prod ‘ yum update -y mapr-\\*’

6. ################EBF #####################

clush -b -g prod “ mkdir -p /root/patch/ebf/”

clush -b -g prod “ chmod 755 /root/patch/ebf/”

clush -bg pord –copy /root/patch/05Feb2019 –dest /root/patch/ebf/

yum local install /root/patch/ebf/<filename>.rpm

· rm /opt/mapr/conf/mapruserticket

· /opt/mapr/server/configure.sh -R

· systemctl start mapr-warden

· Copied the newly generated mapruserticket to all other nodes except edge nodes

· Make sure the file owner is mapr:maprg

· /opt/mapr/server/configure.sh -R

· Regenerate of mapr service ticket with impersonation in POSIX client. We need to make sure that map service ticket has impersonation “true”. Then run the below configure. Also update the fuse.conf with new ticket.

maprlogin generateticket -type servicewithimpersonation -user mapr -out /opt/mapr/conf/mapr\_impersonation

/opt/mapr/server/configure.sh -N rdiprod1 -u mapr -g maprg -C esekilx5643.rnd.ki.sw.ericsson.se:7222,esekilx5644.rnd.ki.sw.ericsson.se:7222,esekilx5645.rnd.ki.sw.ericsson.se:7222 -Z esekilx5640.rnd.ki.sw.ericsson.se:5181,esekilx5641.rnd.ki.sw.ericsson.se:5181,esekilx5642.rnd.ki.sw.ericsson.se:5181 -c -secure

https://mapr.com/docs/61/UpgradeGuide/RestartingClusterServices.html

7. Configure the Ecosystem components especially Spark, Livy, Hive, Impala, Hue, Oozie. Gp02 should be given special care for R libraries. We need to test Log Monitoring and Metrics Monitoring Modules. Let us copy the .xml of respective node to the newly created conf.

- ################EBF #####################

clush -b -g prod “ mkdir -p /root/patch/ebf/”

clush -b -g prod “ chmod 755 /root/patch/ebf/”

clush -bg pord –copy /root/patch/05Feb2019 –dest /root/patch/ebf/

yum local install /root/patch/ebf/<filename>.rpm

**MIGRATION OF MAPR NODES FROM ORACLE JAVA TO OPENJDK**

Effective Environment: Production Cluster (butters)

Issue which led to this action: The production cluster was facing Drill Issue where Drill was crashing due to OutOfMemory error dumping lots of Core Files in almost each node. MapR Support Team recommended to have same Java Version on all nodes throughout the cluster.

Completion Date: 17th May 2019 (JIRA Ticket: <https://wcdma-jira.rnd.ki.sw.ericsson.se/browse/MC-468> )

Priority in which the Java requires to be upgraded:

1. Data Nodes (butters01-23, butters31-48)
2. Data Node with Running ResourceManager (butters30)
3. Data Node with Standby ResourceManager (butters29)
4. Management Node with Running Ecosystem Components (cartman05)
5. Management Node with Standby Ecosystem Components (cartman04)
6. Slave CLDB & Follower Zookeeper Node (cartman01 & cartman03)
7. Master CLDB & Leader Zookeeper Node (cartman02)

\*\* Perform manual CLDB failover to get the cldbmaster to another node. Perform zookeeper restart to get the zookeeper leader to another node.

1. Edge nodes (kenny01 and kenny02)
2. Client nodes (kenny05 and kenny06)

Process/Steps Taken in each node:

**Case1: Node currently installed with Oracle Java (java-1.8.0\_144)**

1. Check the current version of java and java processes running:

* [root@cartman05 ~]# java -version

java version "1.8.0\_144"

Java(TM) SE Runtime Environment (build 1.8.0\_144-b01)

Java HotSpot(TM) 64-Bit Server VM (build 25.144-b01, mixed mode)

* [root@cartman05 ~]# ps -ef | grep java

(Output should show process running with java-1.8.0\_144 version)

1. Install OpenJDK at the node:

[root@cartman05 ~]# yum install -y java-1.8.0-openjdk-devel.x86\_64

1. Now check the java version (which should have updated with new OpenJDK version):

[root@cartman05 ~]# java -version

openjdk version "1.8.0\_212"

OpenJDK Runtime Environment (build 1.8.0\_212-b04)

OpenJDK 64-Bit Server VM (build 25.212-b04, mixed mode)

\*\*But java processes will be running with the old java version at present.

1. Check the file: /etc/profile

The file should be appended with/updated with following 3 statements:

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk

export JRE\_HOME=/usr/lib/jvm/java-1.8.0-openjdk/jre

export PATH=$JAVA\_HOME/bin:$PATH

\*\* If not updated automatically, kindly add above 3 statements manually to the file /etc/profile.

1. (Optional: If req) For the zookeeper nodes, restart zookeeper service:

[root@cartman05 ~]# systemctl restart mapr-zookeeper

1. Restart warden, so that Java process picks up the new java version:

[root@cartman05 ~]# systemctl restart mapr-warden

1. Check if the Java process reflects the new version of Java:

[root@kenny02 ~]# ps -ef | grep java

mapr     20659     1  0 05:20 ?        00:00:16 /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7\_6.x86\_64/jre/bin/java -XX:ErrorFile………………………….

**Case2: Node currently installed with OpenJDK Java (java-1.8.0\_181/191)**

In the nodes, which are already installed with lower version of OpenJDK, an extra step is required before the zookeeper and warden restart, which is (\*\*Select the option with the latest installed package):

[root@cartman01 ~]# update-alternatives --config java

There are 2 programs which provide 'java'.

  Selection    Command

-----------------------------------------------

   1           /usr/java/jdk1.8.0\_144/jre/bin/java

\*+ 2           java-1.8.0-openjdk.x86\_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7\_6.x86\_64/jre/bin/java)

Enter to keep the current selection[+], or type selection number: 2

[OPTIONAL] We can also remove the Oracle Java from the nodes: “rm -rf /usr/java” (But, we have not performed this step and the Oracle Java still resides in all the nodes at /usr/java/jdk1.8.0\_144 which is not functional)

**UPGRADATION OF MANA SANDBOX CLUSTER**

**(from MapR6.0.1 TO MapR6.1.0)**

**Method used:**Offline Manual Upgrade (Offline)

**Date of Upgradation:**15th November 2018

**CONSIDERATIONS BEFORE UPGRADATION:**

* After upgrading MapR Core to MapR 6.1.0, upgrade ecosystem components to a MEP6.0.0. This must be done before enabling MapR 6.1.0 features.
* The offline upgrade procedure requires an outage of the entire cluster.

**UPGRADATION STEPS:**

1. **Understand MapR Core/MEP Dependencies:**

1.a. [Operating System Support Matrix (MapR 6.x)](https://mapr.com/docs/home/InteropMatrix/r_os_matrix_6.x.html) -

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OS Version / MapR Version | MapR 6.1.0 | MapR 6.0.1 | MapR 6.0.0 | MapR 5.2.2 |
| CentOS7.3.1611 (64bit) | Yes | Yes | Yes | No |

\*\*Check cluster’s OS version at file /etc/redhat-release

|  |  |
| --- | --- |
| This MapR Core Version | Supports These MEPs |
| 6.1.0 | 6.0.0 |
| 6.0.1 | 5.0.1, 5.0.0 |

1.b. [MEP Support by MapR Core Version](https://mapr.com/docs/61/InteropMatrix/r_mep_support_core_version.html) –

1.c. Component Versions for Released MEPs –

        \*\* Refer to the excel made: Matrix for components in Sandbox (Tab MapR6.0.1)

1. **Plan for MapR Core Upgrade:**

2.a. [Upgrading and Your License](https://mapr.com/docs/61/UpgradeGuide/License-upgrade.html#license-upgrade) –

If upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in cluster: skip the step as we are upgrading from version 6.0.1 to 6.1.0.

Current Licenses in cluster:

* MapR M7 Edition
* MapR Enterprise Edition
* MapR Base Edition
* Base MapR POSIX Client for fast secure file access

NOTE: \*\* **MapR Metering Feature:**Beginning with the MapR 6.1 release, MapR software supports metering. Annual subscriptions will continue to be offered, but metering gives you the option of purchasing a variable consumption plan that is based on usage.

Ref: <https://mapr.com/docs/61/UpgradeGuide/License-upgrade.html#license-upgrade>

2.b. [Upgradation & Installation Considerations](https://mapr.com/docs/61/UpgradeGuide/License-upgrade.html#license-upgrade) –

        Ref: <https://mapr.com/docs/61/ReleaseNotes/install_upgrade_notes.html>

|  |  |
| --- | --- |
| Upgrading to MapR 6.1.0 Might Require an OS Upgrade | Our cluster use OS as CentOS7.3.1611 which supports MapR6.1.0 |
| Data-on-wire-encryption | Beginning with MapR 6.1, data-on-wire encryption is enabled by default for newly created volumes on secure clusters. Data-on-wire encryption is *not* supported for non-secure clusters. |
| MapR 6.1.0 and MEP 6.0.0 | MapR 6.1.0 requires MEP 6.0.0. MEP 3.0.1 or later can coexist with MapR 6.1.0 only temporarily in the context of an upgrade. |
| Metrics Monitoring | MapR 6.1.0 requires a minimal level of metrics monitoring to be configured to support metering. If metrics monitoring is already configured before the upgrade, you must upgrade it as part of the MapR Expansion Pack upgrade. |
| Regenerating the mapruserticket File | There is no “mapruser” file at /opt/mapr/conf to be regenerated. |

2.c. Planning Your MapR Core Upgrade –

JDK version present: 1.8.0\_161 (this is compatible with MapR6.1.0)

\*\*Volume mirroring from a lower MapR version to higher MapR version is supported. For example, you can mirror volumes from a MapR 4.0.1 cluster to a MapR 5.2 cluster.

1. **Plan for the MEP Upgrade:**

Install ecosystem components as part of a MEP. You will be offered packs to install that contain selected component versions. After upgrading, you may want to upgrade to a more recent MEP to get the latest patch releases or newer versions of ecosystem components.

Most MapR core versions support multiple MEPs, but the set of ecosystem components that you run in the cluster must all belong to the same MEP. When you upgrade a MEP, all components are replaced with the versions contained in the newly selected MEP.

1. **Perform Pre-upgrade steps for MapR Core:**

Ref - <https://mapr.com/docs/61/UpgradeGuide/Preparing-to-Upgrade.html#PreparingtoUpgrade>

4.a. Verify System Requirements for All nodes:

* Software dependencies – We are using package manager, where we specify a repository that contains the dependency package(s), and allow the package manager to automatically install them when you upgrade the MapR packages.
* Hardware requirements –

CPU: x86\_64 (checked) [command: uname -m]

OS: CentOS Linux release 7.3.1611 (checked) [command: lsb\_release -a]

Memory: minimum req 16GB (checked) [command: free -g]

Disk: space to /tmp & /opt directory

DNS: hostname (checked) [command: hostname -f & ping]

Users: common users (checked) [same UID of “mapr” user across the cluster]

Java: 1.8.0\_161 (checked) [command: java -version]

Others: NTP running, No syslog enabled (all checked)

4.b. Design Health Checks:

* Check for alerts in the cluster –

Command: maprcli node list -columns svc (checked; all OK)

* Non-trivial test – not required as we have no data in Sandbox Cluster

4.d. Backup Configuration Files:

Creation  of env\_override.sh file to store custom settings for environmental variables not required as no custom settings done in env.sh. Upgrading to a new MapR release causes the env.sh file to be replaced and removes any custom settings.

Ref: <https://mapr.com/docs/61/ReferenceGuide/env_override.sh.html#concept_hnz_4yd_mdb>

4.e. Migrate from Apache Hbase: (not required)

Ref: <https://mapr.com/docs/61/UpgradeGuide/Preparing-to-Upgrade.html#PreparingtoUpgrade>

1. **Prepare to Upgrade MEP Components:**

5.a. Pre-Upgrade Steps for Drill: (ike01-05)

* Backup taken: (at /root/mapr\_6.0.1\_backup/drill\_1.13.0\_backup/)

             /opt/mapr/drill/drill-1.13.0/conf

             /opt/mapr/drill/drill-1.13.0/jars

* Backup of Storage Plugins: done

5.b. Pre-Upgrade Steps for HBase Client (ike01 & 02): backup of /opt/mapr/hbase/hbase-1.1.8/conf taken at /root/mapr\_6.0.1\_backup/hbase\_1.1.8\_backup/.

5.c. Pre-Upgrade Steps for Hive: (ike04 & 05)

* Backup the metastore database: mysqldump -u root -p hive -r hive\_dump.sql (done)
* Backup taken of: /opt/mapr/hive/hive-2.1/conf/ & /opt/mapr/hive/hive-2.1/lib/ at

 \*For a major version update (for example, Hive-2.1-1803 to Hive-2.3-1808), user configuration from a previous version is **only** copied to a folder with an old version timestamp but not copied to a new version conf folder.

5.d. Pre-Upgrade Steps for HttpFS: (ike04 & 05)

Backup taken of: /opt/mapr/httpfs/httpfs-1.0/etc/hadoop/ & /opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/

5.e. Pre-Upgrade Steps for Hue: (ike04 & 05)

* Create a Hue database dump as a JSON object:

source /opt/mapr/hue/hue-3.12.0/build/env/bin/activate

(env)[root@ike04 ~]# hue dumpdata > ~/dump-hue-3.12.0.json

* Backup taken of: /opt/mapr/hue/hue-3.12.0/desktop/conf
* Backup taken of: /opt/mapr/livy/ livy-0.3.0/conf

5.f. Pre -Upgrade Steps for MapR Monitoring:

* Backup taken of:

**All nodes**

/opt/mapr/conf/conf.d/warden.collectd.conf

/opt/mapr/collectd/collectd-5.7.2/etc/collectd.conf

/etc/logrotate.d/collectd

**ike01**

/opt/mapr/conf/conf.d/warden.grafana.conf

/opt/mapr/grafana/grafana-4.6.1/etc/grafana/grafana.ini

/opt/mapr/grafana/grafana-4.6.1/etc/grafana/ldap.toml

**ike02**

/opt/mapr/conf/conf.d/warden.opentsdb.conf

/opt/mapr/opentsdb/opentsdb-2.4.0/etc/opentsdb/opentsdb.conf

/opt/mapr/opentsdb/opentsdb-2.4.0/etc/opentsdb/logback.xml

opt/mapr/opentsdb/opentsdb-2.4.0/bin/tsdb\_cluster\_mgmt.sh

5.g. Pre-Upgrade Steps for Oozie: (ike04 & 05)

* Stop any jobs or coordinators that are in a RUNNING or SUSPENDED state.
* Backup taken of: /opt/mapr/oozie/oozie-4.3.0/conf
* Backup taken of oozie database: mysqldump -u root -p oozie -r oozie\_dump.sql (done)
* Backup of the old share libraries and examples from the following directories: (done)
* maprfs:///oozie/share

5.h. Pre-Upgrade Steps for Spark: (all nodes)

* Backup taken of: /opt/mapr/spark/spark-2.2.1/conf

\*\* Backup taken of: /etc/my.cnf

1. **Setting up Repositories:**

This has been already done in Sandbox Cluster.

Ref: <https://mapr.com/docs/61/UpgradeGuide/SetUpInternetRepoRHEL.html>

1. **Offline & Manual Upgrade Procedure:**

7.a. Send mail to stakeholders about upgrade.

        Disable Puppet: clush -w ike[01-05] puppet agent –disable

7.b. CLDB nodes: ike01 & ike02(master)

       Zookeeper nodes: ike01(follower), ike02(leader) & ike03(follower)

7.c. Ensure no MapR processes are running:

        ps -ef | grep mapr | grep -v grep | awk '{print $2}' | xargs kill

         pkill -u mapr (Check using ps -ef|grep mapr)

7.d. Upgrade MapR core packages by installing the appropriate MapR package key.

      rpm --import <http://package.mapr.com/releases/pub/maprgpg.key>

7.e. Stop Warden on CLDB nodes, then remaining & also zookeeper:

        clush -w ike[01-02] systemctl stop mapr-warden

        clush -w ike[03-04] systemctl stop mapr-warden

        clush -w ike[01-03] systemctl stop mapr-zookeeper

7.f. Remove existing patches: rpm -e mapr-patch (Check using rpm -qa mapr-patch)

7.g. ike01:

yum update mapr-cldb mapr-core mapr-core-internal mapr-fileserver mapr-hadoop-core mapr-mapreduce2 mapr-nfs mapr-nodemanager mapr-zookeeper mapr-zk-internal mapr-ericsson

ike02:

yum update mapr-cldb mapr-core mapr-core-internal mapr-fileserver mapr-hadoop-core mapr-mapreduce2 mapr-nfs mapr-nodemanager mapr-zookeeper mapr-zk-internal mapr-ericsson

ike03:

yum update mapr-core mapr-core-internal mapr-fileserver mapr-hadoop-core mapr-mapreduce2 mapr-nfs mapr-nodemanager mapr-resourcemanager mapr-webserver mapr-apiserver mapr-zookeeper mapr-zk-internal mapr-ericsson

ike04:

yum update mapr-core mapr-core-internal mapr-fileserver mapr-hadoop-core mapr-historyserver mapr-mapreduce2 mapr-nfs mapr-nodemanager mapr-resourcemanager mapr-webserver mapr-apiserver mapr-ericsson

ike05:

yum update mapr-core mapr-core-internal mapr-fileserver mapr-hadoop-core mapr-historyserver mapr-mapreduce2 mapr-nfs mapr-nodemanager mapr-ericsson

7.h. Check /opt/mapr/MapRBuildVersion contains the expected value:6.0.1.20180404222005.GA(now)

1. **Upgrade the MEP Components:**

8.a. ike01:

yum update mapr-drill mapr-hbase mapr-collectd mapr-grafana mapr-kafka mapr-spark mapr-asynchbase

ike02:

yum update mapr-drill mapr-hbase mapr-collectd mapr-opentsdb mapr-kafka mapr-spark mapr-asynchbase

ike03:

yum update mapr-drill mapr-collectd mapr-spark

ike04:

yum update mapr-drill mapr-hive mapr-hiveserver2 mapr-hivemetastore mapr-hivewebhcat mapr-hue mapr-livy mapr-collectd mapr-oozie mapr-oozie-internal mapr-spark mapr-spark-historyserver

ike05:

yum update mapr-drill mapr-hive mapr-hiveserver2 mapr-hivemetastore mapr-hivewebhcat mapr-hue mapr-livy mapr-collectd mapr-kafka mapr-oozie mapr-oozie-internal mapr-spark mapr-spark-historyserver mapr-fluentd

**\*\*HttpFS:**(Do not upgrade httpfs using yum update, as it will create 2 folders within /opt/mapr/httpfs which can cause ambiguity in which folder to pick up. Hence, remove the httpfs package and reinstall it on respective node which will create only 1 folder name “httpfs-1.0” under /opt/mapr/httpfs)

yum remove mapr-httpfs

rm -rf /opt/mapr/httpfs/

yum install mapr-httpfs

<https://mapr.com/docs/61/UpgradeGuide/Upgrading-HttpFS.html>

1. **Perform Post-Upgrade Steps for MEP**

9.a. Drill: (all nodes)

i. Reapply custom changes: (/opt/mapr/drill/drill-<version>/conf)

* drill-override.conf
* drill-env.sh
* drill-distrib.conf

ii. Run /opt/mapr/server/configure.sh -R

iii. Issue “jps” command to check Drillbit running

iv. Check in MCS if Drill is running

v. Verify storage plugin configurations at <https://192.4.25.203:7047/storage>

Can access the logs at: /opt/mapr/drill/drill-<version>/logs/drillbit.log.

9.b. Hbase Client: (ike01 & 02)

Merge HBase Client configuration files from with the new default files in /opt/mapr/hbase/hbase-<version>/conf/. Be sure not to simply copy over the configuration files: to avoid overwriting the default files, conduct a merge.

9.c. Hive: (ike04 & 05)

i. Migrate any custom conf to /opt/mapr/hive/hive-2.3/conf/

ii. Update Hive Metastore: /opt/mapr/hive/hive-2.3/bin/schematool -dbType mysql -upgradeSchema

iii. Run /opt/mapr/server/configure.sh -R

iv. Verify metastore database update completed successfully. Run the show tables command in Hive and make sure it returns a complete list of all your Hive tables.

9.d. HttpFS: (ike04 & 05)

i. Migrate any custom configuration settings in:

* /opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/webapps/webhdfs/WEB-INF/web.xml
* /opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/server.xml
* /opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/tomcat-users.xml
* /opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-site.xml

9.e. Hue: (ike04 & 05)

i. Migrate required changes to /opt/mapr/hue/hue-<version>/desktop/conf/hue.ini

ii. Update database schema:

source /opt/mapr/hue/hue-4.2.0/bin/activate

hue syncdb --noinput

hue migrate --merge

deactivate

        iii. maprcli node services -name hue -action restart -nodes 192.4.25.204 192.4.25.205

        iv. Transfer custom configuration for Livy at /opt/mapr/livy/livy-<version>/conf/

        iii. Re-check the number of tables in Hue

9.f. MapR Monitoring:

Add customized properties from the configuration files backed up before the upgrade to the files in the new installation directories. Run /opt/mapr/server/configure.sh -R

9.g. Oozie: (ike04 & 05)

i. Add customized configuration at /opt/mapr/oozie/oozie-<version>/conf/

ii. If Oozie installation is configured to use MySQL database and upgrading to a new Oozie version, copy the JDBC driver jar file for MySQL to following directory: /opt/mapr/oozie/oozie-<oozie version>/libext

iii. If it is present, remove the old warden.oozie.conf under the /opt/mapr/conf/conf.d/ directory before running the configure.sh -R command.

iv. Run /opt/mapr/server/configure.sh -R

v. Stop the Oozie service: maprcli node services -name oozie -action stop -nodes ike04 ike05

vi. Upgrade database schema: /opt/mapr/oozie/oozie-<version>/bin/ooziedb.sh upgrade -run

vii. Start the Oozie service: maprcli node services -name oozie -action start -nodes ike04 ike05

viii. If needed, update oozie shared libraries

       (Ref: <https://mapr.com/docs/61/Oozie/UpdatingOozieSharedLibs.html#task_erj_jyr_3z> )

9.h. Spark: (all nodes)

i. Migrate any custom configuration settings to /opt/mapr/spark/spark-<version>/conf

ii. If previously configured Spark to use the Spark JAR file from a location on the MapR Filesystem, you need to copy the latest JAR file to the MapR Filesystem and reconfigure the path to the JAR file in the spark-defaults.conf file (<https://mapr.com/docs/61/Spark/ConfigureSparkJARLocation.html#ConfigureSparkJARLocation> )

iii. If Spark SQL is configured to work with Hive, copy hive-site.xml file into the conf directory (/opt/mapr/spark/spark-<version>/conf).

iv. Run configure.sh -R.

v. Delete the old Spark directory from /opt/mapr/spark.

1. **Perform Post-Upgrade Steps for MapR Core:**

**Step 1-**

10.a. clush -w ike[01-03] service mapr-zookeeper start

10.b. clush -w ike[01-02] service mapr-warden start

10.c. clush -w ike[03-05] service mapr-warden start

10.d. Set new cluster version: maprcli config save -values {mapr.targetversion:"`cat /opt/mapr/MapRBuildVersion`"}

10.e. Verify new cluster version: maprcli config load -keys mapr.targetversion

10.f. Check all services working:

maprcli node list -columns hostname,csvc

maprcli node cldbmaster

/opt/mapr/initscripts/zookeeper qstatus

**Step 2 –**

1. On all nodes, manually merge new configuration settings from the /opt/mapr/conf.new/warden.conf file into the /opt/mapr/conf/warden.conf file.
2. On all nodes, manually merge new configuration settings from the files in the /opt/mapr/conf/conf.d.new/ directory to the files in the /opt/mapr/conf/conf.d/ directory. /opt/mapr/conf/conf.d.new/ directory only had warden.nodemanager.conf which matched as same present at /opt/mapr/conf/conf.d/ directory.
3. Manually merge the port and authentication configuration information in the /opt/mapr/conf/web.conf directory from the pre-6.0 MapR version to the /opt/mapr/apiserver/conf/properties.cfg file of the upgraded MapR version. (for ike03 & ike04)
4. To update the Base License, copy the new Base License file from the /opt/mapr/conf.new/ directory to the /opt/mapr/conf/ directory on every node in your cluster.
5. Enable new features: Check the new features using **maprcli cluster feature list** command and enable it using **maprcli cluster feature enable -all**
6. Enable Puppet: clush -w ike[01-05] puppet agent --enable

**Post Upgrade Steps for MapR Core**

Step 1 (Restart and Check Cluster Services)

1. No custom edits present at /opt/mapr/conf/env.sh file (tallied /opt/mapr/conf/env.sh and /opt/mapr/conf/env.sh.2018-11-15.02-15). Hence no merge was required. **Checked for ike01/**
2. Set new cluster version:

maprcli config save -values {mapr.targetversion:"`cat /opt/mapr/MapRBuildVersion`"}

Check using:

maprcli config load -keys mapr.targetversion

Step 2 (Manually Update Configuration Files)

1. Manually merge new configuration settings from the /opt/mapr/conf.new/warden.conf file into the /opt/mapr/conf/warden.conf file.

Step 3 (Manually Update your License)

Copy the new Base License file from the /opt/mapr/conf.new/ directory to the /opt/mapr/conf/ directory on every node in your cluster. (cp /opt/mapr/conf.new/BaseLicense.txt /opt/mapr/conf/

)

**UPGRADATION OF MANA PRODUCTION CLUSTER**

**(from MapR6.0.1 TO MapR6.1.0)**

**Method used:**Manual Rolling Upgrade (Online)

**Date of Upgradation:**11th December 2018

**CONSIDERATIONS BEFORE UPGRADATION:**

* In rolling upgrade, the ecosystem components will continue to work as long as the ecosystem components are not updated. After upgrading MapR Core to MapR 6.1.0, you must upgrade ecosystem components to MEP 6.0.0 or later, and this must be done before you enable MapR 6.1 features.
* In a manual rolling upgrade, you upgrade the MapR software one node at a time so that the cluster remains operational throughout the process.

**UPGRADATION STEPS:**

1. **Understand MapR Core/MEP Dependencies:**

1.a. [Operating System Support Matrix (MapR 6.x)](https://mapr.com/docs/home/InteropMatrix/r_os_matrix_6.x.html) -

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OS Version / MapR Version | MapR 6.1.0 | MapR 6.0.1 | MapR 6.0.0 | MapR 5.2.2 |
| CentOS7.3.1611 (64bit) | Yes | Yes | Yes | No |

\*\*Check cluster’s OS version at file /etc/redhat-release

|  |  |
| --- | --- |
| This MapR Core Version | Supports These MEPs |
| 6.1.0 | 6.0.0 |
| 6.0.1 | 5.0.1, 5.0.0 |

1.b. [MEP Support by MapR Core Version](https://mapr.com/docs/61/InteropMatrix/r_mep_support_core_version.html) –

1.c. Component Versions for Released MEPs –

        \*\* Refer to the excel made: Matrix for components in Sandbox (Tab MapR6.0.1)

1. **Plan for MapR Core Upgrade:**

2.a. [Upgrading and Your License](https://mapr.com/docs/61/UpgradeGuide/License-upgrade.html#license-upgrade) –

If upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in cluster: skip the step as we are upgrading from version 6.0.1 to 6.1.0.

Current Licenses in cluster:

* MapR M7 Edition
* MapR Enterprise Edition
* MapR Base Edition
* Base MapR POSIX Client for fast secure file access
* MapR POSIX Client for fast secure file access

NOTE: \*\* **MapR Metering Feature:**Beginning with the MapR 6.1 release, MapR software supports metering. Annual subscriptions will continue to be offered, but metering gives you the option of purchasing a variable consumption plan that is based on usage.

Ref: <https://mapr.com/docs/61/UpgradeGuide/License-upgrade.html#license-upgrade>

2.b. [Upgradation & Installation Considerations](https://mapr.com/docs/61/UpgradeGuide/License-upgrade.html#license-upgrade) –

        Ref: <https://mapr.com/docs/61/ReleaseNotes/install_upgrade_notes.html>

|  |  |
| --- | --- |
| Upgrading to MapR 6.1.0 Might Require an OS Upgrade | Our cluster use OS as CentOS7.3.1611 which supports MapR6.1.0 |
| Data-on-wire-encryption | Beginning with MapR 6.1, data-on-wire encryption is enabled by default for newly created volumes on secure clusters. Data-on-wire encryption is *not* supported for non-secure clusters. |
| MapR 6.1.0 and MEP 6.0.0 | MapR 6.1.0 requires MEP 6.0.0. MEP 3.0.1 or later can coexist with MapR 6.1.0 only temporarily in the context of an upgrade. |
| Metrics Monitoring | MapR 6.1.0 requires a minimal level of metrics monitoring to be configured to support metering. If metrics monitoring is already configured before the upgrade, you must upgrade it as part of the MapR Expansion Pack upgrade. |
| Regenerating the mapruserticket File | There is no “mapruser” file at /opt/mapr/conf to be regenerated. |

2.c. Planning Your MapR Core Upgrade –

JDK version present: 1.8.0\_161 (this is compatible with MapR6.1.0)

\*\*Volume mirroring from a lower MapR version to higher MapR version is supported. For example, you can mirror volumes from a MapR 4.0.1 cluster to a MapR 5.2 cluster.

1. **Plan for the MEP Upgrade:**

Install ecosystem components as part of a MEP. You will be offered packs to install that contain selected component versions. After upgrading, you may want to upgrade to a more recent MEP to get the latest patch releases or newer versions of ecosystem components.

Most MapR core versions support multiple MEPs, but the set of ecosystem components that you run in the cluster must all belong to the same MEP. When you upgrade a MEP, all components are replaced with the versions contained in the newly selected MEP.

1. **Perform Pre-upgrade steps for MapR Core:**

Ref - <https://mapr.com/docs/61/UpgradeGuide/Preparing-to-Upgrade.html#PreparingtoUpgrade>

4.a. Verify System Requirements for All nodes:

* Software dependencies – We are using package manager, where we specify a repository that contains the dependency package(s), and allow the package manager to automatically install them when you upgrade the MapR packages.
* Hardware requirements –

CPU: x86\_64 (checked) [command: uname -m]

OS: CentOS Linux release 7.3.1611 (checked) [command: lsb\_release -a]

Memory: minimum req 16GB (checked) [command: free -g]

Disk: space to /tmp & /opt directory

DNS: hostname (checked) [command: hostname -f & ping]

Users: common users (checked) [same UID of “mapr” user across the cluster]

Java: 1.8.0\_161 (checked) [command: java -version]

Others: NTP running, No syslog enabled (all checked)

4.b. Design Health Checks:

* Check for alerts in the cluster –

Command: maprcli node list -columns svc (kept the output as backup at butters01)

* Non-trivial test – done

4.c. Pause Cross-Cluster Operations:

Check for kenny01

4.d. Backup Configuration Files:

Creation  of env\_override.sh file to store custom settings for environmental variables not required as no custom settings done in env.sh. Upgrading to a new MapR release causes the env.sh file to be replaced and removes any custom settings.

Fuse.conf backup is not required as fuse.conf.backup file gets created during upgradation.

Ref: <https://mapr.com/docs/61/ReferenceGuide/env_override.sh.html#concept_hnz_4yd_mdb>

4.e. Migrate from Apache Hbase: (not required)

Ref: <https://mapr.com/docs/61/UpgradeGuide/Preparing-to-Upgrade.html#PreparingtoUpgrade>

1. **Prepare to Upgrade MEP Components:**

\*\*Create folder to keep backup:

clush -w butters[01-23] mkdir /root/mapr\_6.0.1\_backup

clush -w butters[29-38] mkdir /root/mapr\_6.0.1\_backup

clush -w butters[40-48] mkdir /root/mapr\_6.0.1\_backup

clush -w cartman[01-05] mkdir /root/mapr\_6.0.1\_backup

clush -w kenny[01-02] mkdir /root/mapr\_6.0.1\_backup

clush -w kenny[05-06] mkdir /root/mapr\_6.0.1\_backup

5.a. Pre-Upgrade Steps for Drill: (butters01-23, 29-38, 40-48)

* Backup taken: (at /root/mapr\_6.0.1\_backup/drill\_1.13.0\_backup/)

             /opt/mapr/drill/drill-1.13.0/conf

             /opt/mapr/drill/drill-1.13.0/jars

* Backup of Storage Plugins: done

5.b. Pre-Upgrade Steps for HBase Client (butters01-03, butters29, butters46-48, cartman05 & kenny02): backup of /opt/mapr/hbase/hbase-1.1.8/conf taken at /root/mapr\_6.0.1\_backup/hbase\_backup/.

5.c. Pre-Upgrade Steps for Hive: (butters29, cartamn04 & 05)

* Backup the metastore database: mysqldump -u root -p hive -r hive\_dump.sql (done at cartman05 /root)
* Backup taken of: /opt/mapr/hive/hive-2.1/conf/ at  /root/mapr\_6.0.1\_backup/hive\_conf.

 \*For a major version update (for example, Hive-2.1-1803 to Hive-2.3-1808), user configuration from a previous version is **only** copied to a folder with an old version timestamp but not copied to a new version conf folder.

5.d. Pre-Upgrade Steps for HttpFS: (cartman04 & 05)

Backup taken of: /opt/mapr/httpfs/httpfs-1.0/etc/hadoop/ at /root/mapr\_6.0.1\_backup/httpfs\_conf

5.e. Pre-Upgrade Steps for Hue: (cartman04 & 05)

* Create a Hue database dump as a JSON object: (done at cartman04 & 05 /root)

source /opt/mapr/hue/hue-3.12.0/build/env/bin/activate

(env)[root@ike04 ~]# hue dumpdata > ~/dump-hue-3.12.0.json

* Backup taken of: /opt/mapr/hue/hue-3.12.0/desktop/conf at /root/mapr\_6.0.1\_backup/hue\_conf
* Backup taken of: /opt/mapr/livy/ livy-0.3.0/conf

5.f. Pre -Upgrade Steps for MapR Monitoring:

* Backup taken of:

**All nodes except kenny05&kenny06**

/opt/mapr/conf/conf.d/warden.collectd.conf

/opt/mapr/collectd/collectd-5.7.2/etc/collectd.conf

/etc/logrotate.d/collectd

**butters48**

/opt/mapr/conf/conf.d/warden.grafana.conf

/opt/mapr/grafana/grafana-4.6.1/etc/grafana/grafana.ini

/opt/mapr/grafana/grafana-4.6.1/etc/grafana/ldap.toml

**butters01-03**

/opt/mapr/conf/conf.d/warden.opentsdb.conf

/opt/mapr/opentsdb/opentsdb-2.4.0/etc/opentsdb/opentsdb.conf

/opt/mapr/opentsdb/opentsdb-2.4.0/etc/opentsdb/logback.xml

opt/mapr/opentsdb/opentsdb-2.4.0/bin/tsdb\_cluster\_mgmt.sh

5.g. Pre-Upgrade Steps for Oozie: (cartman04 & 05)

* Stop any jobs or coordinators that are in a RUNNING or SUSPENDED state.
* Backup taken of: /opt/mapr/oozie/oozie-4.3.0/conf at /root/mapr\_6.0.1\_backup/Oozie\_conf
* Backup taken of oozie database: mysqldump -u root -p oozie -r oozie\_dump.sql (done)
* Backup of the old share libraries and examples from the following directories:
* maprfs:///oozie/share

5.h. Pre-Upgrade Steps for Spark: (butters01-23, 29-38, 40-48, cartman04-05, kenny02)

* Backup taken of: /opt/mapr/spark/spark-2.2.1/conf

\*\* Backup taken of: /etc/my.cnf (etc\_my\_cnf)

\*\*Backup taken of : /etc/Hadoop (Hadoop\_backup)

1. **Setting up Repositories:**

Ref: <https://mapr.com/docs/61/UpgradeGuide/SetUpLocalRepoRHEL.html>

\*\* Repository was already downloaded from internet at /var/www/html/yum/base

Updated the symlinks of mapr\_core & mapr\_eco:

ln -sfn /var/www/html/yum/base/v6.1.0 mapr\_core

ln -sfn /var/www/html/yum/base/MEP/MEP-6.0 mapr\_eco

1. Run yum clean all in all nodes of cluster.
2. Take note of current and upgraded version of all services from all nodes (use yum info)

1. **Perform the Manual Rolling Upgrade:**

**Manual Rolling Upgrade Description:**

Group Upgrade Order: -

1 – cartman01,03(slave CLDB) & 02(master CLDB)

2 – cartman04 (to maintain zookeeper quorum)

2 – butters40,41,42 & cartman05 (upgrade mapr-gateway nodes before fileserver)

3 – upgrade butters29(standby resourcemanger) before butters30(active resourcemanager)

4 – butters[01-23], butters[31-38], butters[43-48] (upgrade fileserver nodes)

**Manual Rolling Upgrade Procedure:**

7.a. Send mail to stakeholders about upgrade.

7.b. Disable Puppet:

        clush -g dnall puppet agent --disable

 7.c. Kill the yarn applications:

        yarn application -list

        yarn application -kill <ApplicationId>

7.d. Stop posix-client service on Posix nodes (kenny01,05,06)

        Stop NFS on all nodes (from MCS)

        Stop RM on all nodes (from MCS)

        Stop NM on all nodes (from MCS)

7.e. Stop Warden on all nodes, CLDB nodes & also zookeeper:

        clush -w butters[01-23] systemctl stop mapr-warden

        clush -w butters[29-38] systemctl stop mapr-warden

        clush -w butters[40-48] systemctl stop mapr-warden

        clush -w cartman[04-05] systemctl stop mapr-warden

        clush -w kenny02 systemctl stop mapr-warden

        clush -w cartman[01-03] systemctl stop mapr-warden (CLDB nodes last)

        clush -w cartman[02-04] systemctl stop mapr-zookeeper

7.f. Ensure no MapR processes are running:

        clush -ab 'ps -ef | grep mapr | grep -v grep |wc -l'

       clush -g dnall ps -ef | grep mapr | grep -v grep | awk '{print $2}' | xargs kill

       ps -ef | grep mapr| grep -v grep | awk '{system("kill -9 "$2)}'

7.g. Remove existing patches: rpm -e mapr-patch (Check using rpm -qa mapr-patch)

       \*\*Patch present on all nodes (except cartman01); kenny05 & 06 has mapr-patch-client

       clush -ab ‘yum erase -y <mapr-patch-name>’

7.h. Upgrade the MapR core and MEP components:

        clush -g dnall "yum update mapr-\\*"

7.i. Configure the node:

       clush -g dnall /opt/mapr/server/configure.sh -R

7.j. Start Zookeeper:

       clush -w cartman[02-04] systemctl stop mapr-zookeeper

       clush -w cartman[02-04] systemctl daemon-reload

       clush -w cartman[02-04] ‘jps’ (check with this command)

7.k. Update the configuration change for Drill and Spark in Puppet for upgradation (done by Avijit)

7.l. Enable Puppet: clush -g dnall puppet agent --enable

7.m. Applied patches from MapR SFTP location: core, patch-client, patch-posix-client-basic (v6.1)

clush -ab "yum localinstall -y /root/patch\_11122018/mapr-patch-6.1.0.20180926230239.GA-20181129115411.x86\_64.rpm"

7.n. Start Warden:

clush -w cartman[01-03] systemctl start mapr-warden (CLDB nodes)

         clush -g dnall systemctl start mapr-warden

        clush -w kenny[05-06] service mapr-posix-client-basic start

7.o. Check that the CLDB is running. If output is displayed, the CLDB is running. If not, start CLDB.

        maprcli node list

 7.p. Wait for the containers to synchronize, run the following command, and check that there is no output:

          /opt/mapr/server/mrconfig info containers resync local

         No output signifies that the containers are synchronized.

1. **Perform Post-Upgrade Steps for MEP**

8.a. Drill:

i. Reapply custom changes: (/opt/mapr/drill/drill-<version>/conf)

* drill-override.conf
* drill-env.sh
* drill-distrib.conf

ii. Run /opt/mapr/server/configure.sh -R

iii. Issue “jps” command to check Drillbit running

iv. Check in MCS if Drill is running

v. Verify storage plugin configurations at <https://192.4.25.203:7047/storage>

Can access the logs at: /opt/mapr/drill/drill-<version>/logs/drillbit.log.

8.b. Hbase Client:

Merge HBase Client configuration files from with the new default files in /opt/mapr/hbase/hbase-<version>/conf/. Be sure not to simply copy over the configuration files: to avoid overwriting the default files, conduct a merge.

8.c. Hive:

i. Migrate any custom conf to /opt/mapr/hive/hive-2.3/conf/

ii. Update Hive Metastore: /opt/mapr/hive/hive-2.3/bin/schematool -dbType mysql -upgradeSchema

iii. Run /opt/mapr/server/configure.sh -R

iv. Verify metastore database update completed successfully. Run the show tables command in Hive and make sure it returns a complete list of all your Hive tables.

8.d. HttpFS:

i. Migrate any custom configuration settings in:

* /opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/webapps/webhdfs/WEB-INF/web.xml
* /opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/server.xml
* /opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/tomcat-users.xml
* /opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-site.xml

8.e. Hue:

i. Migrate required changes to /opt/mapr/hue/hue-<version>/desktop/conf/hue.ini

ii. Update database schema:

source /opt/mapr/hue/hue-4.2.0/bin/activate

hue syncdb --noinput

hue migrate --merge

deactivate

        iii. maprcli node services -name hue -action restart -nodes 192.4.25.204 192.4.25.205

        iv. Transfer custom configuration for Livy at /opt/mapr/livy/livy-<version>/conf/

        iii. Re-check the number of tables in Hue

8.f. MapR Monitoring:

Add customized properties from the configuration files backed up before the upgrade to the files in the new installation directories. Run /opt/mapr/server/configure.sh -R

8.g. Oozie:

i. Add customized configuration at /opt/mapr/oozie/oozie-<version>/conf/

ii. If Oozie installation is configured to use MySQL database and upgrading to a new Oozie version, copy the JDBC driver jar file for MySQL to following directory: /opt/mapr/oozie/oozie-<oozie version>/libext

iii. If it is present, remove the old warden.oozie.conf under the /opt/mapr/conf/conf.d/ directory before running the configure.sh -R command.

iv. Run /opt/mapr/server/configure.sh -R

v. Stop the Oozie service: maprcli node services -name oozie -action stop -nodes ike04 ike05

vi. Upgrade database schema: /opt/mapr/oozie/oozie-<version>/bin/ooziedb.sh upgrade -run

vii. Start the Oozie service: maprcli node services -name oozie -action start -nodes ike04 ike05

viii. If needed, update oozie shared libraries

       (Ref: <https://mapr.com/docs/61/Oozie/UpdatingOozieSharedLibs.html#task_erj_jyr_3z> )

8.h. Spark:

i. Migrate any custom configuration settings to /opt/mapr/spark/spark-<version>/conf

ii. If previously configured Spark to use the Spark JAR file from a location on the MapR Filesystem, you need to copy the latest JAR file to the MapR Filesystem and reconfigure the path to the JAR file in the spark-defaults.conf file (<https://mapr.com/docs/61/Spark/ConfigureSparkJARLocation.html#ConfigureSparkJARLocation> )

iii. If Spark SQL is configured to work with Hive, copy hive-site.xml file into the conf directory (/opt/mapr/spark/spark-<version>/conf).

iv. Run configure.sh -R.

v. Delete the old Spark directory from /opt/mapr/spark.

1. **Perform Post-Upgrade Steps for MapR Core:**

**Step 1-**

9.a. clush -w ike[01-03] service mapr-zookeeper start

9.b. clush -w ike[01-02] service mapr-warden start

9.c. clush -w ike[03-05] service mapr-warden start

9.d. Set new cluster version: maprcli config save -values {mapr.targetversion:"`cat /opt/mapr/MapRBuildVersion`"}

9.e. Verify new cluster version: maprcli config load -keys mapr.targetversion

9.f. Check all services working:

maprcli node list -columns hostname,csvc

maprcli node cldbmaster

/opt/mapr/initscripts/zookeeper qstatus

**Step 2 –**

1. On all nodes, manually merge new configuration settings from the /opt/mapr/conf.new/warden.conf file into the /opt/mapr/conf/warden.conf file.
2. On all nodes, manually merge new configuration settings from the files in the /opt/mapr/conf/conf.d.new/ directory to the files in the /opt/mapr/conf/conf.d/ directory. /opt/mapr/conf/conf.d.new/ directory only had warden.nodemanager.conf which matched as same present at /opt/mapr/conf/conf.d/ directory.
3. Manually merge the port and authentication configuration information in the /opt/mapr/conf/web.conf directory from the pre-6.0 MapR version to the /opt/mapr/apiserver/conf/properties.cfg file of the upgraded MapR version. (for ike03 & ike04)
4. To update the Base License, copy the new Base License file from the /opt/mapr/conf.new/ directory to the /opt/mapr/conf/ directory on every node in your cluster.
5. Enable new features: Check the new features using **maprcli cluster feature list** command and enable it using **maprcli cluster feature enable -all**

**Set RM logs in MapR NFS**

<https://mapr.com/support/s/article/How-to-change-Nodemanager-local-dirs-to-MAPR-NFS?language=en_US>

<property>

<name>yarn.nodemanager.local-dirs</name>

<value>/mapr/MapRDev/var/mapr/local/${mapr.host}/nm-local-dir</value>

</property>

**Installation spark 2.4.0 & spark 2.3.3**

I have used the below command to install spark:

**rpm -ivh --replacefiles --force --prefix=/ mapr-spark-2.3.3.100.201905170600-1.noarch.rpm  
after this please execute configure.sh -R to integrate the new spark with MapR core.**

You can now submit spark-job using below command:

2.3.3:

=======  
/opt/mapr/spark/spark-2.3.3/bin/spark-submit --master yarn --deploy-mode client --class org.apache.spark.examples.SparkPi /opt/mapr/spark/spark-2.3.3/examples/jars/spark-examples\_2.11-2.3.3.100-mapr-611.jar  
  
2.4.0:

=======  
/opt/mapr/spark/spark-2.4.0/bin/spark-submit --master yarn --deploy-mode client --class org.apache.spark.examples.SparkPi /opt/mapr/spark/spark-2.4.0/examples/jars/spark-examples\_2.11-2.4.0.0-mapr-620.jar



MAPR-NFS

====

#clush -bg dsldev 'cp /opt/mapr/conf/mapr\_fstab.sample /opt/mapr/conf/mapr\_fstab'

#clush -bg dsldev 'chown mapradmin:mapr /opt/mapr/conf/mapr\_fstab'

#clush -bg dsldev 'echo localhost:/mapr /mapr nolock,nfsvers=3 >> /opt/mapr/conf/mapr\_fstab'

#clush -bg dsldev 'mkdir /mapr'

Mapr-Spark installation without MEP

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Install spark-2.4

rpm -ivh --replacefiles --force --prefix=/ mapr-spark-2.4.0.0.201905170634-1.noarch.rpm

2.

Create a zip archive containing all the JARs from the SPARK\_HOME/jars directory.

For example:

cd /opt/mapr/spark/spark-2.4.0/jars/

zip /opt/mapr/spark/spark-2.4.0/spark-jars.zip ./\*

Copy the zip file from the local file system to a world-readable location on MapR-FS. You can upload it to the home of the current user:

hadoop fs -put /opt/mapr/spark/spark-2.4.0/spark-jars.zip

For example:

hadoop fs -put /opt/mapr/spark/spark-2.4.0/spark-jars.zip /user/mapr/

3.

a.

Chown the spark-2.4

b.

modify the spark.env.sh to version spark.2.4

c.

Set the spark.yarn.archive property in the spark-defaults.conf file to point to the world-readable location where you added the zip file. Apply this setting on the node where you will be submitting your Spark jobs.

spark.yarn.archive maprfs:///<path to zip>

For example:

spark.yarn.archive maprfs:///user/mapr/spark-jars.zip

4. Testing

/opt/mapr/spark/spark-2.4.0/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode client  /opt/mapr/spark/spark-2.4.0/examples/jars/spark-examples\_2.11-2.4.0.0-mapr-620.jar 10

/opt/mapr/spark/spark-2.4.0/bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode cluster  /opt/mapr/spark/spark-2.4.0/examples/jars/spark-examples\_2.11-2.4.0.0-mapr-620.jar 10

$$$$$$$$$$$$$$$

RStudio

========

URL: <https://rstudio.com/products/rstudio/download-server/redhat-centos/>

wget <https://download2.rstudio.org/server/centos6/x86_64/rstudio-server-rhel-1.2.5001-x86_64.rpm>

sudo yum install rstudio-server-rhel-1.2.5001-x86\_64.rpm

Set parameter in Conf file

--------------

vi /etc/rstudio/rserver.conf

# Server Configuration File

www-port=8788

www-address=esekilx5571.rnd.ki.sw.ericsson.se

To access with PAM user

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

#cd /etc/pam.d/

->make symlink to login to rstudio

# ls -s login rstudio or ls -s /etc/pam.d/login /etc/pam.d/rstudio

Start rstudio

-------------

#/usr/sbin/rstudio-server start

#/usr/sbin/rstudio-server status

# rstudio-server restart

Now access in browser:

----------------------

<http://esekilxgp02.rnd.ki.sw.ericsson.se:8788/auth-sign-in>

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

1.Need to add below parametter:

/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop

<property>

    <name>yarn.resourcemanager.webapp.address</name>

    <value>esekilx5642.rnd.ki.sw.ericsson.se:8090</value>

  </property>

<property>

    <name>yarn.resourcemanager.webapp.address</name>

    <value>esekilx5642.rnd.ki.sw.ericsson.se:8090,esekilx5641.rnd.ki.sw.ericsson.se:8090,esekilx5640.rnd.ki.sw.ericsson.se:8090</value>

  </property>

2.To connect with spark through Rstudio

install and set the below commands:

library(sparklyr)

Sys.setenv("SPARK\_HOME" = "/opt/mapr/spark/spark-2.3.1")

Sys.setenv(YARN\_CONF\_DIR = "/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop")

config <- spark\_config()

sc <- spark\_connect(master = "yarn-client", app\_name = "SparklyR\_RL\_Returners", config = config)

spark\_disconnect(sc)

cat /sys/block/sdj/device/state

:

cat /dev/null >  filename

truncate -s 0 file