Lab: HDFS Management

Managing & Using HDFS

1. Table of Contents

2. HDFS File Management

The goal of this lab is to demonstrate HDFS file management through HDFS CLI.

Lets upload a file into HDFS

1. Login to Cockpit and ssh to admin@edge.cluster

```
ssh admin@edge.cluster
```

2. Create a file, call it 'hello.txt', with some text content in it

```
echo "Hello World" > /tmp/hello.txt
```

3. Lets create a directory in HDFS

   hdfs dfs -mkdir /user/admin/demo

4. Lets put it to HDFS

```
hdfs dfs -put /tmp/hello.txt /user/admin/demo/
```

Lets run some read activities

1. See the contents of /user/admin and /user/admin/demo

```
hdfs dfs -ls /user/admin
hdfs dfs -ls /user/admin/demo
```

2. See the contents of hello.txt

```
hdfs dfs -cat /user/admin/demo/hello.txt
```

3. Lets do a checksum of hello.txt

```
hdfs dfs -checksum /user/admin/demo/hello.txt
```

4. Lets see the size usage of hello.txt

```
hdfs dfs -du /user/admin/demo/hello.txt
```

5. lets download hello.txt to local

```
hdfs dfs -get /user/admin/demo/hello.txt
cat hello.txt
```

6. lets change permission of /user/admin

```
    hdfs dfs -ls -d /user/admin
    hdfs dfs -chmod og-rwx /user/admin
    hdfs dfs -ls -d /user/admin
```

7. Lets delete hello.txt and its directory

```
    hdfs dfs -rm /user/admin/demo/hello.txt
    hdfs dfs -rm /user/admin/demo/
```

3. HDFS Access Control

The goal of this lab is to demonstrate HDFS ACL command line.

Before we can use HDFS ACL, we need to enable it. HDP does not enable this by default neither expose the configuration as standard settings as the recommended manner is to use Ranger.

1. In Ambari HDFS service configuration, under Custom hdfs-site, add a new property: dfs.namenode.acls.enabled=true
2. Save and restart necessary services

Lets try granting access using ACL

1. Login to cockpit and ssh to edge.cluster as user 'admin'

```
ssh admin@edge.cluster
```

2. Lets give 'jeff' read & write access to /user/admin

```
hdfs dfs -getfacl /user/admin
hdfs dfs -setfacl -m user:jeff:rw- /user/admin
hdfs dfs -getfacl /user/admin
```

3. switch to user 'jeff' (create the user if its not there yet), note that he can write to /user/admin

```
su - jeff
hdfs dfs -mkdir /user/admin/jeff-demo
hdfs dfs -rmdir /user/admin/jeff-demo
exit
```

4. Lets revoke it back

```
hdfs dfs -getfacl /user/admin
hdfs dfs -setfacl -x user:jeff /user/admin
hdfs dfs -getfacl /user/admin
```

4. HDFS Access Control With Ranger

The goal of this lab is to demonstrate HDFS ACL through Ranger. This assumes you already have a user called 'jeff' in Ambari, created by earlier tutorials.

1. Login to Ambari as admin
2. Go to Ranger service page, under Quick Links, click Ranger Admin UI
3. Login with following credentials
   - Username: admin
   - Password: admin
4. Under HDFS component, click the current cluster's HDFS service
   - SandboxCluster_hadoop
5. Click **Add New Policy**
6. Create a policy:
   - policy name: allow jeff to access admin home
   - resource path: /user/admin
   - select user: jeff
   - permission: read-write
7. Click **Add**
8. Login as user jeff in Ambari or cli, jeff should be able to add write into /user/admin/

5. HDFS Snapshot

The goal of this lab is to demonstrate the process of creating snapshot:

1. Before creating snapshot, we need to allow snapshot to a directory. Login to cockpit and SSH to root@edge.cluster

```
ssh root@edge.cluster
```

2. Allowing snapshot can only be done by superuser, lets switch user to hdfs, and then allow snapshot to /user/admin

```
su - hdfs
hdfs dfsadmin -allowSnapshot /user/admin
exit
```

3. Switch to user admin, and lets create a snapshot

```
su - admin
hdfs dfs -ls /user/admin
hdfs dfs -createSnapshot /user/admin
```

4. Note the path to the snapshot that was created, lets list down its content

```
hdfs dfs -ls $SNAPSHOTPATH
```

5. Lets delete the snapshot

```
hdfs dfs -deleteSnapshot /user/admin $SNAPSHOTNAME
```

6. Configuring Heterogeneous Storage

The goal of this lab is to demonstrate the configuration and usage of heterogeneous storage in HDFS. Every docker node in the VM have been configured with a ramdisk at /mnt/ramdisk, so we are going to configure hdfs to use the disk

Adding ramdisk into the pool of directories.

1. Login to Ambari as user 'admin'
2. In HDFS settings page, under Datanode Directory settings, add this entry in the comma separated directory list:

```
[RAM_DISK]/mnt/ramdisk
```

3. Restart necessary services

Setting a HDFS storage policy for a directory

1. Login into Cockpit and ssh to admin@edge.cluster

   ssh admin@edge.cluster

2. Lets create a directory , and set it to use LAZY_PERSIST storage policy. This storage policy is meant to speed up write I/O by accepting the data upload to HDFS into RAM_DISK first, before persisting to DISK storage.

```
hdfs dfs -mkdir /user/admin/lazywrite/
hdfs storagepolicies -listPolicies
hdfs storagepolicies -setStoragePolicy \
    -path /user/admin/lazywrite -policy LAZY_PERSIST
```

3. Lets try to write a file into lazywrite dir

```
echo Hello world > /tmp/hello.txt
time hdfs dfs -put /tmp/hello.txt /user/admin/
time hdfs dfs -put /tmp/hello.txt /user/admin/lazywrite
```

4. Note the difference in time taken. Theoretically LAZY_PERSIST will take shorter time to write a file. As the sample file is small here, it may not be a noticable number.

7. Configuring Centralized Cache

The goal of this lab is to demonstrate the process of configuring HDFS centralized cache.

1. Login to Cockpit and ssh to root@edge.cluster

   ssh root@edge.cluster

2. Lets allow higher memlock for worker1 and worker2 (where datanode reside) (256MB)

```
ssh root@worker1.cluster
echo "* hard memlock 262144" > /etc/security/limits.conf
echo "* soft memlock  262144" > /etc/security/limits.conf
# repeat for worker2.cluster
```

3. Login to Ambari as admin, and then in HDFS settings, add a custom hdfs-site property (192MB):

```
dfs.datanode.max.locked.memory=201326592
```

4. Restart affected datanodes and HDFS services
5. Switch back to cockpit
6. Lets create a 128MB cache pool for user hive to use

```
su - hdfs
hdfs cacheadmin -addPool HivePool \
   -owner hive -group hive -mode 0770 -limit 134217728 -maxTtl never
exit
```

7. Switch to hive user

```
su - hive
```

8. Lets cache a hive directory

```
hdfs cacheadmin -addDirective -path /apps/hive/warehouse/zomato \
    -pool HivePool -ttl never
```

9. check the status of the caches

```
hdfs cacheadmin -listDirectives -stats
```

10. Lets run a query in hive, and check the cache status.

```
beeline -u jdbc:hive2://master1.cluster:10000/default \
      -n hive -e "select * from zomato"
hdfs cacheadmin -listDirectives -stats
```

8. Configuring Quota

The goal of this lab is to demonstrate the process of creating a name and space quota for directories.

1. Login to cockpit and ssh to root@edge.cluster
2. Lets set a quota of 100MB for /user/admin

```
su - hdfs
hdfs dfsadmin -setSpaceQuota 104857600 /user/admin
exit
```

3. Let see the quota statistics

```
su - admin
hdfs dfs -count -q -h /user/admin
exit
```

4. Lets clear the quota

```
su - hdfs
hdfs dfsadmin -clrSpaceQuota /user/admin
hdfs dfs -count -q -h /user/admin
exit
```

9. Configuring Rack Awareness

The goal of this lab is to enable rack awareness for HDFS

1. Login to Ambari as user admin
2. Go to Hosts tab
3. Click on a Host (eg: worker1.cluster)
4. Click Host Actions > Set Rack
5. Key in /rack1
6. Restart necessary services
7. On HDFS services overview page, click Heatmap tab, you will see now that worker1.cluster is on a different rack

10. Datanode Decommissioning & Recommissioning

The goal of this lab is to demonstrate the process of decommissioning and recommissioning datanodes

1. Login to Ambari as user admin
2. Go to Hosts tab
3. Click on a Worker Host (eg: worker2.cluster)
4. On Datanode service, click the drop down button beside it, and click Decommission
   - Note: if your datanode is the only one, you cant decomission it
5. Datanode should be in the state of decomissioning now, wait until it change state to decommissioned
   - After decommissioned, datanode can be safely taken offline for maintenance
6. To recommission, follow the same steps, but select Recommission on the service button

11. Replicating Data With DistCP

The goal of this lab is to demonstrate copying data using DistCP

1. Login to Cockpit and ssh to admin@edge.cluster
2. Using the steps from creating snapshot, create a snapshot of /user/admin
3. Lets copy the snapshot to a different directory

```
hdfs dfs -mkdir /tmp/mycopy
hadoop distcp hdfs://master1.cluster:8020/$SNAPSHOTPATH \
 hdfs://master1.cluster:/tmp/mycopy/
hdfs dfs -ls /tmp/mycopy
```

12. Setting Up Namenode HA

This lab is to demonstrate the process of setting up Namenode HA

1. Login to Ambari as user admin
2. Click on the Zookeeper service, then click on Service Action > Add Zookeeper Server
3. Add enough zookeeper to make a quorum of 3. (master1, master2, master3)
4. Go to the HDFS service configuration page, and click Service Actions > Enable NameNode HA
5. Provide a nameservice ID. Eg: trainingcluster
6. Next, assign host for journal node and standby namenode.
   - Standby node: master2
   - Journalnode: master1, master2, master3
7. Review configuration changes and click Next
8. Execute manual step requested by Ambari, click Next, and wait for installation
9. Execute manual step requested by Ambari (initialize journalnode), click next and wait for component start
10. Execute manual step requested by Ambari (initialize HA metadata), click next and wait for HA finalization
11. Login to Cockpit and ssh to admin@edge.cluster

```
ssh admin@edge.cluster
```

12. List HDFS to see if Namenode is operational correctly

```
hdfs dfs -ls /
```