

HADOOP: HDFS

Aspectos esenciales de HDFS	1
Configuración de HDFS	2
El flujo de operaciones de lectura/escritura en HDFS	3
Entender la interfaz de usuario de <i>namenode</i>	5
Explorando los comandos de HDFS	8
Comandos para administrar HDFS	10

En este tema, profundizaremos en el componente HDFS (sistema de archivos) de Apache Hadoop y cubriremos los siguientes temas:

- Aspectos esenciales del sistema de archivos distribuidos de Hadoop
- El flujo operativo de lectura/escritura en HDFS
- Exploración de los comandos de HDFS

Aspectos esenciales de HDFS

HDFS es un sistema de archivos distribuido que ha sido diseñado para funcionar sobre un clúster con hardware estándar. La arquitectura de HDFS es tal que no hay necesidad específica de hardware de alta gama. HDFS es un sistema altamente tolerante a los fallos y puede manejar los fallos de los nodos en un clúster sin pérdida de datos. El objetivo principal del diseño de HDFS es servir archivos de datos de gran tamaño de forma eficiente. HDFS logra esta eficiencia y un alto rendimiento en la transferencia de datos al permitir el acceso a los datos en el sistema de archivos.

Las siguientes son las características importantes de HDFS:

- **Tolerancia a fallos:** Muchos ordenadores que trabajan juntos como un clúster necesariamente van a tener fallos de hardware. Los fallos de hardware, como los fallos de disco, los problemas de conectividad de la red y los fallos de la memoria RAM, podrían interrumpir el procesamiento y causar un importante tiempo de inactividad. Esto podría conducir a la pérdida de datos y al incumplimiento de los acuerdos de nivel de servicio críticos. HDFS está diseñado para resistir estos fallos de hardware detectando las fallas y tomando las acciones de recuperación necesarias.

Los datos en HDFS se dividen entre las máquinas del clúster como trozos de datos llamados bloques. Estos bloques se replican en varias máquinas del clúster para que haya redundancia. Así, incluso si un nodo/máquina se vuelve completamente inutilizable y se apaga, el procesamiento puede continuar con la copia de los datos presentes en los nodos donde los datos fueron replicados.

- **Streaming de datos:** El acceso en flujo permite que los datos se transfieran en forma de flujo constante y continuo. Esto significa que, si los datos de un archivo en HDFS necesitan ser procesados, HDFS comienza a enviar los datos a medida que lee el archivo y no espera a que se lea todo el archivo. El cliente que está consumiendo estos datos comienza a procesar los datos inmediatamente, ya que recibe el flujo de HDFS. Esto hace que el procesamiento de datos sea realmente rápido.

- **Almacenamiento de grandes volúmenes:** HDFS se utiliza para almacenar grandes volúmenes de datos. HDFS funciona mejor cuando los archivos de datos almacenados son grandes, en lugar de tener un gran número de archivos pequeños. El tamaño de los archivos en la mayoría de los clústeres de Hadoop oscila entre los gigabytes y los terabytes. El almacenamiento se escala linealmente a medida que se añaden más nodos al clúster.

- **Portable:** HDFS es un sistema altamente portable. Dado que se basa en Java, cualquier máquina o sistema operativo que pueda ejecutar Java debería ser capaz de ejecutar HDFS. Incluso en la capa de hardware, HDFS es flexible y se ejecuta en la mayoría de las plataformas de hardware disponibles. La mayoría de los clústeres a nivel de producción se han montado en hardware básico.

- **Interfaz sencilla:** La interfaz de línea de comandos de HDFS es muy similar a la de cualquier sistema Linux/Unix. Los comandos son similares en la mayoría de los casos. Por lo tanto, si uno se siente cómodo con la realización de acciones básicas de archivo en Linux/Unix, el uso de comandos con HDFS debería ser muy fácil.

Los siguientes dos *demonios* son responsables de las operaciones en HDFS:

- Namenode
- Datanode

Los demonios *namenode* y *datanode* trabajan juntos para almacenar archivos en el clúster. Estos demonios se comunican entre sí a través de TCP/IP.

Configuración de HDFS

Toda la configuración relacionada con HDFS se realiza añadiendo/actualizando las propiedades en el archivo **hdfs-site.xml** que se encuentra en la carpeta `conf` bajo la carpeta de instalación de Hadoop. En HDP 2.6.5 se encuentra en la carpeta `/etc/hadoop/conf`.

Las siguientes son las diferentes propiedades que forman parte del archivo `hdfs-site.xml`:

- `dfs.namenode.rpc-address`: Especifica la dirección RPC de *namenode* única en el clúster. Los servicios/demonios, como los demonios secundarios de *namenode* y *datanode*, utilizan esta dirección para conectarse al demonio de *namenode* siempre que necesiten comunicarse. Esta propiedad se muestra en el siguiente fragmento de código:

```
<property>
  <name>dfs.namenode.rpc-address</name>
  <value>node1.hcluster:8022</value>
</property>
```

- `dfs.namenode.http-address`: Especifica la URL que se puede utilizar para monitorizar el demonio *namenode* desde un navegador. Esta propiedad se muestra en el siguiente fragmento de código:

```
<property>
  <name>dfs.namenode.http-address</name>
```

```
<value>node1.hcluster:50070</value>
```

```
</property>
```

- **dfs.replication:** Especifica el factor de replicación para la replicación de bloques de datos a través de los demonios de *datanode*. El valor predeterminado es 3, como se muestra en el siguiente fragmento de código:

```
<property>
```

```
  <name>dfs.replication</name>
```

```
  <value>3</value>
```

```
</property>
```

dfs.blocksize: Especifica el tamaño del bloque.

En el siguiente ejemplo, el tamaño se especifica en bytes (134.217.728 bytes son 128 MB):

```
<property>
```

```
  <name>dfs.blocksize</name>
```

```
  <value>134217728</value>
```

```
</property>
```

fs.permissions.umask-mode: Especifica el valor de umask que se utilizará al crear archivos y directorios en HDFS.

En el siguiente fragmento de código se muestra esta propiedad:

```
<property>
```

```
  <name>fs.permissions.umask-mode</name>
```

```
  <value>022</value>
```

```
</property>
```

El flujo de operaciones de lectura/escritura en HDFS

Para entender mejor HDFS, necesitamos comprender el flujo de operaciones para los dos escenarios siguientes:

- Cuando un archivo se escribe en HDFS
- Cuando un archivo se lee de HDFS

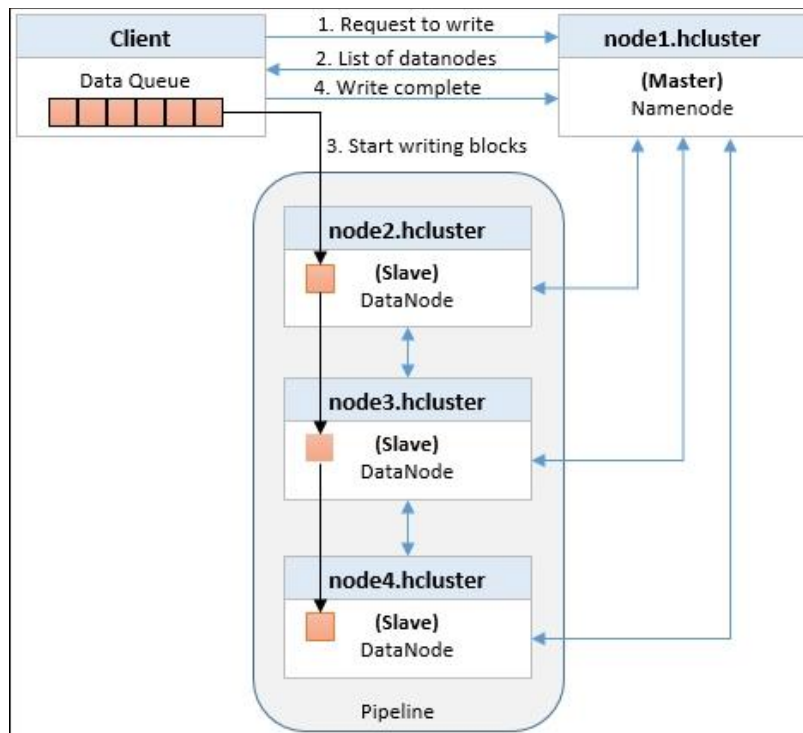
HDFS utiliza un modelo de escritura única y lectura múltiple, donde los archivos se escriben una vez y se leen varias veces. Los datos no pueden ser alterados una vez escritos. Sin embargo, se pueden añadir datos al archivo reabriéndolo. Todos los archivos del HDFS se guardan como bloques de datos.

Escribir archivos en HDFS

La siguiente secuencia de pasos ocurre cuando un cliente intenta escribir un archivo en HDFS:

1. El cliente informa al demonio de *namenode* que quiere escribir un archivo. El demonio de *namenode* comprueba si el archivo ya existe.
2. Si existe, se envía un mensaje apropiado al cliente. Si no existe, el demonio de *namenode* hace una entrada de metadatos para el nuevo archivo.
3. El archivo que se va a escribir se divide en paquetes de datos en el extremo del cliente y se crea una cola de datos. Los paquetes de la cola se transmiten a los *datanodes* del clúster.
4. La lista de *datanodes* viene dada por el demonio *namenode*, que se prepara en función del factor de replicación de datos configurado. Se construye un pipeline para realizar las escrituras a todos los *datanodes* proporcionados por el demonio *namenode*.
5. El primer paquete de la cola de datos se transfiere al primer demonio de *datanode*. El bloque se almacena en el primer demonio de *datanode* y luego se copia al siguiente demonio de *datanode* en el pipeline. Este proceso continúa hasta que el paquete se escribe en el último demonio de *datanode* de la cadena.

El siguiente diagrama muestra el proceso de replicación de bloques de datos a través de los *datanodes* durante una operación de escritura en HDFS:



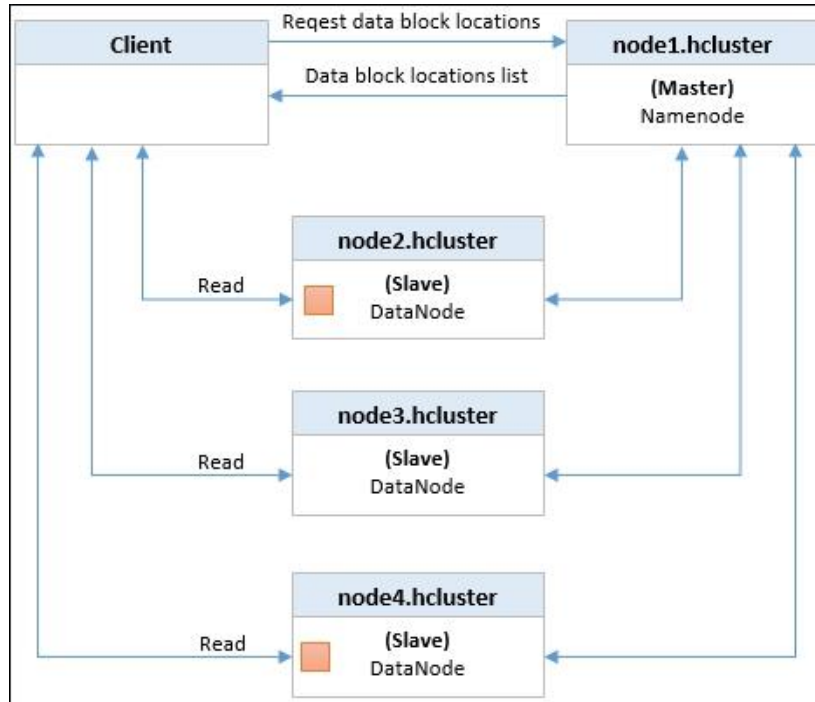
Lectura de archivos en HDFS

Los siguientes pasos ocurren cuando un cliente intenta leer un archivo en HDFS:

1. El cliente se pone en contacto con el demonio *namenode* para obtener la ubicación de los bloques de datos del archivo que quiere leer.
2. El demonio *namenode* devuelve la lista de direcciones de los *datanodes* para los bloques de datos.
3. Para cualquier operación de lectura, HDFS intenta devolver el nodo con el bloque de datos más cercano al cliente. Aquí, lo más cercano se refiere a la proximidad de la red entre el demonio *datanode* y el cliente.

4. Una vez que el cliente tiene la lista, se conecta al demonio del *datanode* más cercano y comienza a leer el bloque de datos usando un *stream*.
5. Una vez leído el bloque por completo, se termina la conexión con el *datanode* y se identifica el demonio del *datanode* que aloja el siguiente bloque de la secuencia y se transmite el bloque de datos. Esto continúa hasta que se lee el último bloque de datos de ese archivo.

El siguiente diagrama muestra la operación de lectura de un archivo en HDFS:

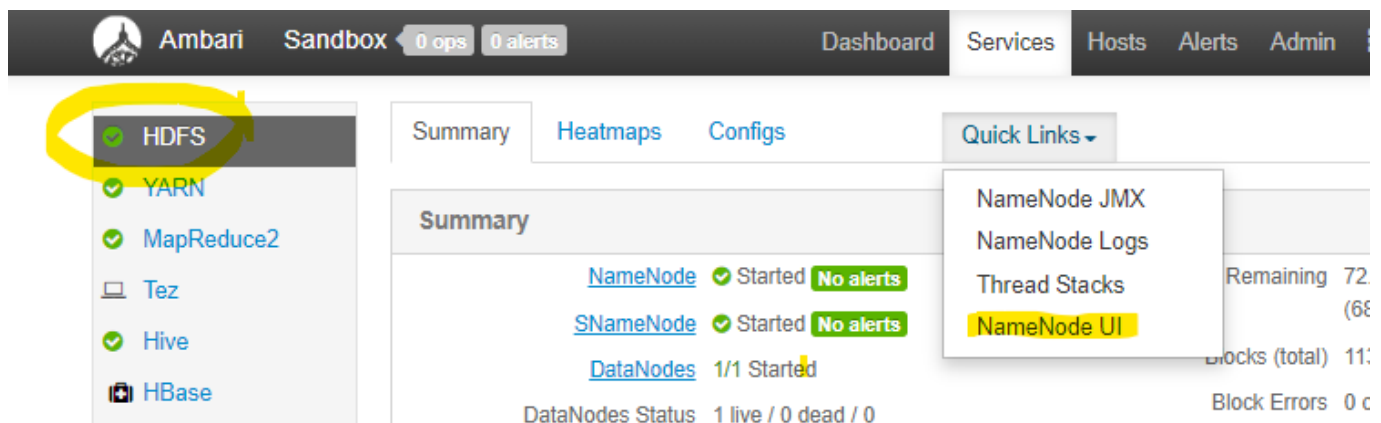


Entender la interfaz de usuario de *namenode*

Hadoop proporciona interfaces web para cada uno de sus servicios. La UI de *namenode* o la interfaz web de *namenode* se utiliza para supervisar el estado del *namenode* y se puede acceder a ella mediante la siguiente URL:

<http://<namenode-server>:50070/>

o desde Ambari:



La UI de *namenode* tiene las siguientes secciones:

- **Información general:** La sección de información general proporciona información básica del *namenode* con opciones para examinar el sistema de archivos y los registros del *namenode*:

Overview 'sandbox-hdp.hortonworks.com:8020' (active)

Started:	Sat Oct 29 11:35:06 UTC 2022
Version:	2.7.3.2.6.5.0-292, r3091053c59a62c82d82c9f778c48bde5ef0a89a1
Compiled:	2018-05-11T07:53Z by jenkins from (detached from 3091053)
Cluster ID:	CID-a915d782-8c97-4ece-9351-cd595a0c9b61
Block Pool ID:	BP-243674277-172.17.0.2-1529333510191

El parámetro Cluster ID muestra el número de identificación del clúster. Este número es el mismo en todos los nodos del cluster. Un *pool* de bloques es un conjunto de bloques que pertenecen a un único *namespace*. El parámetro Block Pool Id se utiliza para segregar los grupos de bloques en caso de que haya múltiples espacios de nombres configurados cuando se utiliza la federación HDFS. En la federación HDFS, se configuran múltiples *namenodes* para escalar el servicio de nombres horizontalmente. Estos *namenodes* están configurados para compartir *datanodes* entre ellos.

Resumen: La siguiente es la captura de pantalla de la sección de resumen del clúster desde la interfaz web de *namenode*:

Summary

Security is off.

Safemode is off.

1526 files and directories, 1139 blocks = 2665 total filesystem object(s).

Heap Memory used 99.83 MB of 240 MB Heap Memory. Max Heap Memory is 240 MB.

Non Heap Memory used 75.14 MB of 77.07 MB Committed Non Heap Memory. Max Non Heap Memory is <unbonded>.

Configured Capacity:	105.98 GB
DFS Used:	1.88 GB (1.77%)
Non DFS Used:	25.7 GB
DFS Remaining:	72.82 GB (68.71%)
Block Pool Used:	1.88 GB (1.77%)
DataNodes usages% (Min/Median/Max/stdDev):	1.77% / 1.77% / 1.77% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	0

En la sección *Resumen*, el primer parámetro está relacionado con la configuración de seguridad del clúster. Si Kerberos (el sistema de autorización y autenticación utilizado en Hadoop) está configurado, el parámetro se mostrará como *Security is on*. Si Kerberos no está configurado, el parámetro se mostrará como *Security is off*. El siguiente parámetro muestra información relacionada con los archivos y bloques del clúster. Junto

con esta información, también se muestra la utilización de la memoria *heap* y no *heap*. Los otros parámetros que se muestran en la sección de Resumen son los siguientes:

- Capacidad configurada: Muestra la capacidad total (espacio de almacenamiento) de HDFS.
- DFS Utilizado: Muestra el espacio total utilizado en HDFS.
- Non DFS Used: Muestra la cantidad de espacio utilizado por otros archivos que no forman parte de HDFS. Este es el espacio utilizado por el sistema operativo y otros archivos.
- DFS Remaining: Muestra el espacio total restante en HDFS.
- Block Pool Used: Muestra el espacio total utilizado por el espacio de nombres actual.
- DataNodes usages% (Min, Median, Max, stdDev): Esto muestra los usos a través de todos los datanodes en el clúster. Esto ayuda a los administradores a identificar los nodos desequilibrados, que pueden ocurrir cuando los datos no se colocan de manera uniforme en los datanodes. Los administradores tienen la opción de reequilibrar los nodos de datos utilizando un equilibrador.

Nodos vivos: Este enlace muestra todos los nodos de datos en el clúster como se muestra en la siguiente captura de pantalla:

The screenshot shows the Hadoop web interface with the 'Datanodes' tab selected. The page title is 'Datanode Information'. Below the title, there are status indicators: 'In service' (green checkmark), 'Down' (red exclamation mark), 'Decommissioned' (orange wrench), and a power icon. The 'In operation' section is active. A table lists the datanodes. The first entry is for a node named 'sandbox-hdp.hortonworks.com:50010 (172.18.0.2:50010)' which is in the 'In service' state. The table columns are: Node, Http Address, Last contact, Last Block Report, Capacity, and Blocks. The Capacity column shows a green bar indicating 105.98 GB used out of a total capacity. The Blocks column shows 1137 blocks.

Node	Http Address	Last contact	Last Block Report	Capacity	Blocks
✓ sandbox-hdp.hortonworks.com:50010 (172.18.0.2:50010)	sandbox-hdp.hortonworks.com:50075	0s	15m	105.98 GB	1137

Nodos muertos: Este enlace muestra todos los *datanodes* que están actualmente en un estado muerto en el clúster. Un estado muerto para un demonio *datanode* es cuando no se está ejecutando o no ha enviado un mensaje de *heartbeat* al namenode demonio. Los *datanodes* no pueden enviar *heartbeats* si existe un problema de conexión de red entre las máquinas que alojan los demonios *datanode* y *namenode*. El intercambio excesivo en la máquina del *datanode* hace que la máquina no responda, lo que también impide que el *datanode* demonio envíe *heartbeats*.

Explorando los comandos de HDFS

<https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/FileSystemShell.html>

Para realizar tareas relacionadas con el sistema de archivos, los comandos comienzan con `hdfs dfs`.

Los comandos del sistema de archivos han sido diseñados para comportarse de forma similar a los comandos correspondientes del sistema de archivos de Unix/Linux.

¿Qué es un URI? URI son las siglas de Uniform Resource Identifier. En los comandos que se enumeran a continuación, observarás el uso de URI para la localización de archivos. La sintaxis URI para acceder a un archivo en HDFS es `hdfs://namenodehost/parent/child/<file>`.

Comandos HDFS más utilizados

Los siguientes son algunos de los comandos HDFS más utilizados:

- `ls`: Este comando lista los archivos en HDFS. La sintaxis del comando `ls` es `hdfs dfs -ls <args>`. La siguiente es la captura de pantalla que muestra un ejemplo del comando `ls`:

```
[root@node1 data]# hdfs dfs -ls /user/root/source/
Found 12 items
-rw-r--r--  3 root root    10576161 2014-05-11 14:18 /user/root/source/file1.txt
-rw-r--r--  3 root root    10570494 2014-05-11 14:18 /user/root/source/file10.txt
-rw-r--r--  3 root root    10553286 2014-05-11 14:18 /user/root/source/file11.txt
-rw-r--r--  3 root root    10580789 2014-05-11 14:18 /user/root/source/file12.txt
-rw-r--r--  3 root root    10571366 2014-05-11 14:18 /user/root/source/file2.txt
-rw-r--r--  3 root root    10561045 2014-05-11 14:18 /user/root/source/file3.txt
-rw-r--r--  3 root root    10553346 2014-05-11 14:18 /user/root/source/file4.txt
-rw-r--r--  3 root root    10555458 2014-05-11 14:18 /user/root/source/file5.txt
-rw-r--r--  3 root root    10562324 2014-05-11 14:18 /user/root/source/file6.txt
-rw-r--r--  3 root root    10556108 2014-05-11 14:18 /user/root/source/file7.txt
-rw-r--r--  3 root root    10548089 2014-05-11 14:18 /user/root/source/file8.txt
-rw-r--r--  3 root root    10587287 2014-05-11 14:18 /user/root/source/file9.txt
[root@node1 data]#
```

- `cat`: La sintaxis del comando `cat` es `hdfs dfs -cat URI [URI ...]`.

La siguiente es una muestra de la salida del comando `cat`:

```
[root@node1 data]# hdfs dfs -cat /user/root/source/file2.txt
[[List of artificial intelligence projects]]

===Planning===

===Other===

==Multipurpose projects==

===Software libraries===
```

- `copyFromLocal`: Este comando copia un archivo/archivos del sistema de archivos local a HDFS. La sintaxis del comando `copyFromLocal` es `hdfs dfs -copyFromLocal <localsrc> URI`. La siguiente es una captura de pantalla que muestra un ejemplo del comando `copyFromLocal`:

```
[root@node1 data]# hdfs dfs -copyFromLocal file1.txt /user/root/files
[root@node1 data]# hdfs dfs -ls /user/root/files
Found 1 items
-rw-r--r--  3 root root    10576161 2014-05-11 15:31 /user/root/files/file1.txt
[root@node1 data]#
```

- copyToLocal: La sintaxis del comando copyToLocal es `hdfs dfs -copyToLocal URI <localdst>`.

La siguiente es una captura de pantalla que muestra un ejemplo del comando copyToLocal:

```
[root@node1 data]# hdfs dfs -copyToLocal /user/root/files/file1.txt file_from_hdfs.txt
[root@node1 data]# ls -ltr file_from_hdfs.txt
-rw-r--r-- 1 root root 10576161 May 11 15:35 file_from_hdfs.txt
[root@node1 data]#
```

- cp: Este comando copia archivos dentro de HDFS.

La sintaxis del comando cp es `hdfs dfs -cp URI [URI ...] <dest>`.

La siguiente es la captura de pantalla que muestra un ejemplo del comando cp:

```
[root@node1 data]# hdfs dfs -cp source/file1.txt dest/
[root@node1 data]# hdfs dfs -ls dest/
Found 1 items
-rw-r--r-- 3 root root 10576161 2014-05-11 15:57 dest/file1.txt
[root@node1 data]#
```

- mkdir: Este comando crea un directorio en HDFS.

La sintaxis del comando mkdir es `hdfs dfs -mkdir <rutas>`.

La siguiente es la captura de pantalla que muestra un ejemplo del comando mkdir:

```
[root@node1 data]# hdfs dfs -mkdir /user/root/target
[root@node1 data]# hdfs dfs -ls /user/root
Found 3 items
drwxr-xr-x - root root 0 2014-05-10 13:01 /user/root/destination
drwxr-xr-x - root root 0 2014-05-10 12:46 /user/root/source
drwxr-xr-x - root root 0 2014-05-10 13:03 /user/root/target
[root@node1 data]#
```

- mv: Este comando mueve archivos dentro de HDFS.

La sintaxis del comando mv es `hdfs dfs -mv URI [URI ...] <dest>`.

La siguiente es la captura de pantalla que muestra un ejemplo del comando mv:

```
[root@node1 data]# hdfs dfs -mv source/file2.txt dest/file2.txt
[root@node1 data]# hdfs dfs -ls dest/
Found 2 items
-rw-r--r-- 3 root root 10576161 2014-05-11 15:57 dest/file1.txt
-rw-r--r-- 3 root root 10571366 2014-05-11 14:18 dest/file2.txt
[root@node1 data]#
```

- rm: Este comando borra los archivos de HDFS.

La sintaxis del comando rm es `hdfs dfs -rm URI [URI ...]`.

La siguiente es la captura de pantalla que muestra un ejemplo del comando rm:

```
[root@node1 data]# hdfs dfs -rm dest/*.txt
14/05/11 16:01:31 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 1440 minutes, Emptier interval = 0 minutes.
Moved: 'hdfs://node1.hcluster:8020/user/root/dest/file1.txt' to trash at: hdfs://node1.hcluster:8020/user/root/.Trash/Current
14/05/11 16:01:31 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 1440 minutes, Emptier interval = 0 minutes.
Moved: 'hdfs://node1.hcluster:8020/user/root/dest/file2.txt' to trash at: hdfs://node1.hcluster:8020/user/root/.Trash/Current
[root@node1 data]#
```

- setrep: Este comando establece el factor de replicación para un archivo en HDFS.

La sintaxis del comando `setrep` es `hdfs dfs -setrep [-R] <ruta>`.

La siguiente es la captura de pantalla que muestra un ejemplo del comando `setrep`:

```
[root@node1 data]# hdfs dfs -setrep -w 4 /user/root/destination/enwiki-20130102-pages-articles.xml-001.txt
Replication 4 set: /user/root/destination/enwiki-20130102-pages-articles.xml-001.txt
Waiting for /user/root/destination/enwiki-20130102-pages-articles.xml-001.txt ... done
```

- `tail`: Este comando muestra el kilobyte final del contenido de un archivo en HDFS.

La sintaxis del comando `tail` es `hdfs dfs -tail [-f] URI`.

La siguiente es la captura de pantalla que muestra un ejemplo del comando `tail`:

```
[root@node1 data]# hdfs dfs -tail /user/root/destination/enwiki-20130102-pages-articles.xml-001.txt
o a colour TV when run in its low or medium resolution (525/625 line 60/50 Hz interlace, even on RGB
ptionally crisp and clear, monitor). These models were known as the 520STM (or 520STM). Later F and

=== STE ===

As originally released in the 520STE/1040STE:

== Models ==
```

Comandos para administrar HDFS

Hadoop proporciona varios comandos para administrar HDFS. Los siguientes son dos de los comandos de administración más utilizados en HDFS:

hdfs balancer: En un clúster, se pueden añadir nuevos *datanodes*. La adición de nuevos *datanodes* proporciona más espacio de almacenamiento para el clúster. Sin embargo, cuando se añade un nuevo *datanode*, éste no tiene ningún archivo. Debido a la adición del nuevo *datanode*, los bloques de datos de todos los *datanodes* están en un estado de desequilibrio, es decir, no están repartidos uniformemente entre los *datanodes*. El administrador puede utilizar el comando `balancer` para equilibrar el clúster. El balanceador puede ser invocado usando este comando.

La sintaxis del comando `balancer` es `hdfs balancer -threshold <threshold>`. Aquí, umbral es el umbral de equilibrio expresado en porcentaje. El umbral se especifica como un valor flotante que va de 0 a 100. El valor del umbral por defecto es 10. El equilibrador intenta distribuir los bloques a los *datanodes* infrautilizados. Por ejemplo, si la utilización media de todos los *datanodes* del clúster es del 50 por ciento, el equilibrador, por defecto, intentará recoger los bloques de los nodos que tengan una utilización superior al 60 por ciento (50 por ciento + 10 por ciento) y moverlos a los nodos que tengan una utilización inferior al 40 por ciento (50 por ciento - 10 por ciento).

hdfs dfsadmin: El comando `dfsadmin` se utiliza para ejecutar comandos administrativos en HDFS.

La sintaxis del comando `dfsadmin` es `hdfs dfsadmin <opciones>`. Vamos a entender algunas de las opciones importantes del comando y las acciones que realizan:

`[-report]`: Esto genera un informe de la información básica del sistema de archivos y las estadísticas.

`[-safemode <enter | leave | get | wait>]`: Este modo seguro es un estado del nodo de nombres en el que no acepta cambios en el espacio de nombres (sólo lectura) y no replica ni borra bloques.

`[-saveNamespace]`: Guarda el estado actual del espacio de nombres en un directorio de almacenamiento y reinicia el registro de ediciones.

`[-rollEdits]`: Obliga a renovar el registro de ediciones, es decir, guarda el estado del registro de ediciones actual y crea un nuevo registro de ediciones para las nuevas transacciones.

`[Restaurar almacenamiento fallido (true|false|check)]`: Permite establecer/desactivar o comprobar el intento de restaurar las réplicas de almacenamiento fallidas.

`[-refreshNodes]`: Esto actualiza el demonio *namenode* con el conjunto de *datanodes* permitido para conectarse al demonio *namenode*.

`[-setQuota <cuota> <nombredeldistrito>...<nombredeldistrito>]`: Establece la cuota (el número de elementos) para el directorio/directorios.

`[-clrQuota <nombredeldirectorio>...<nombredeldirectorio>]`: Borra la cuota establecida para el directorio/directorios.

`[-setSpaceQuota <cuota> <nombredeldirectorio>...<nombredeldirectorio>]`: Establece la cuota de espacio en disco para el directorio/directorios.

`[-clrSpaceQuota <nombredeldirectorio>...<nombredeldirectorio>]`: Borra la cuota de espacio en disco para el directorio/directorios.

`[-refreshserviceacl]`: Actualiza el archivo de política de autorización a nivel de servicio. Más adelante aprenderemos más sobre la autorización.

`[-printTopology]`: Imprime el árbol de los racks y sus nodos tal y como lo informa el demonio de namenodes.

`[-refreshNamenodes datanodehost:port]`: Esto recarga los archivos de configuración para un demonio datanode, deja de servir los grupos de bloques eliminados y comienza a servir nuevos grupos de bloques. Un pool de bloques es un conjunto de bloques que pertenecen a un mismo espacio de nombres. Veremos este concepto un poco más adelante.

`[-deleteBlockPool datanodehost:port blockpoolId [force]]`: Esto borra un pool de bloques de un datanode demonio.

`[-setBalancerBandwidth <ancho de banda>]`: Establece el límite de ancho de banda que debe utilizar el equilibrador. El ancho de banda es el valor en bytes por segundo que el balanceador debe utilizar para el movimiento de bloques de datos.

`[-fetchImage <directorio local>]`: Obtiene el último archivo *fsimage* de namenode y lo guarda en el directorio local especificado.

`[-help [cmd]]`: Muestra la ayuda para el comando dado o todos los comandos si no se especifica un comando.