

HDFS

Neftalí Valdez Martínez

OBJETIVOS

- Entender el diseño y operación de Hadoop Distributed File System (HDFS).
- Conocer los aspectos clave de HDFS tales como replica de bloques, Safe Mode, Rack awareness, Alta Disponibilidad, Federación, Respaldo, Snapshots, NFS y HDFS Web UI.
- Ejecución de comandos básicos de HDFS.
- Ejemplos de programas HDFS en Java y C.

LINUX COMANDOS BÁSICOS

- ¿Dónde estoy?
- pwd print working directory
- ¿Cómo moverme de directorio?
- cd change directory
- cd ..
- cd ../..

LINUX COMANDOS BÁSICOS

- Contenido del directorio
- ls
- Contenido de un directorio en específico
- ls /usr
- Long listing
- ls -l
- Todos los archivos
- ls -a

LINUX COMANDOS BÁSICOS

- Crear un directorio
- `mkdir nuevodirectorio`
- Crear un directorio con subdirectorios
- `mkdir -p project/linux/src/assets`
- Crear un archivo
- `touch newfile`
- Borrar un archivo
- `rm file`

LINUX COMANDOS BÁSICOS

- Borrar un archivo preguntando antes de borrar
- `rm -i file`
- Borrar un directorio vacío
- `rm -d dirvacío`
- Borrar un directorio de forma recursiva
- `rm -rf dirconarchivos`

LINUX COMANDOS BÁSICOS

- Mover archivos a un directorio
- `mv file /tmp`

- Renombrar
- `mv file frenombrado`

LINUX COMANDOS BÁSICOS

- Copiar archivos
- `cp file filebackup`
- Copiar archivo a otro directorio
- `cp file /backup`
- Copiar un directorio con todos sus archivos
- `cp -r Pictures /opt/backup`

DISEÑO DE HDFS

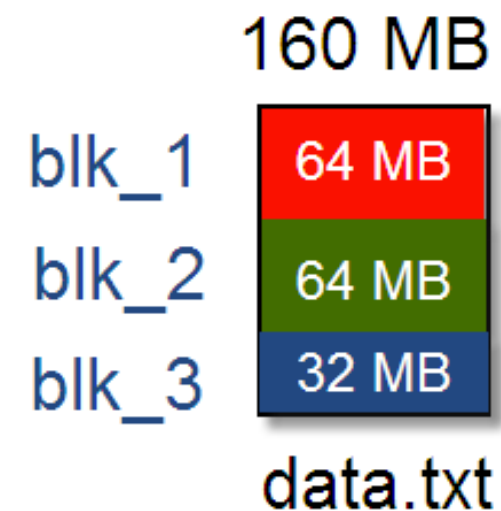
- HDFS fue diseñado para procesamiento Big Data.
- Puede soportar muchos usuarios
- No fue creado como un sistema de archivos en paralelo, relaja muchos de los requerimientos de concurrencia y coherencia.
- El diseño HDFS fue pensado para escribir una vez y leer muchas veces, restringe la escritura a sólo un usuario.
- Los Bytes son escritos al final del stream, y se garantiza el orden de escritura.



DISEÑO HDFS

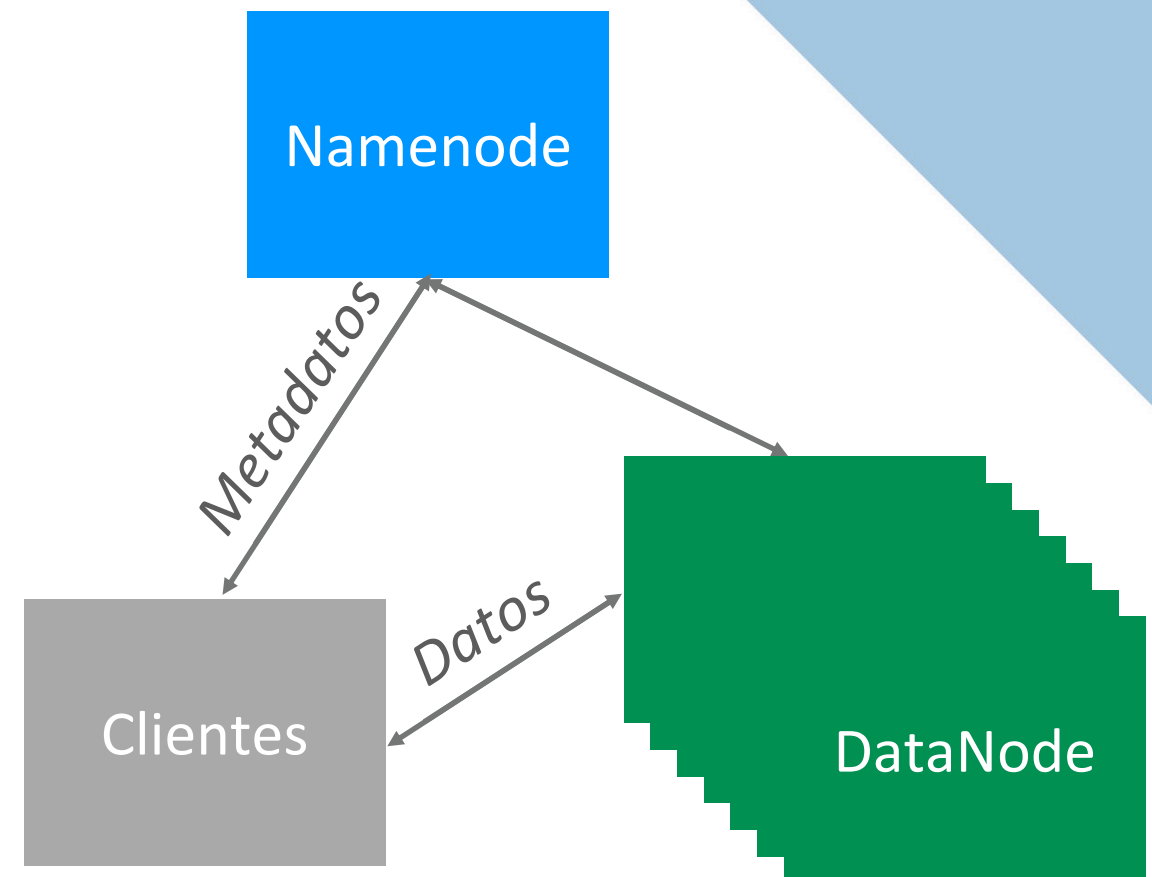
- HDFS está diseñado para la lectura de grandes cantidades de datos en bruto que llegan mediante streaming.
- El bloque de HDFS típico es de 64MB o 128 MB.
- El aspecto clave en HDFS es la localidad de los datos acorde a la filosofía de mover el procesamiento, no los datos.
- Mantiene múltiples copias de los datos dentro del cluster.

HDFS



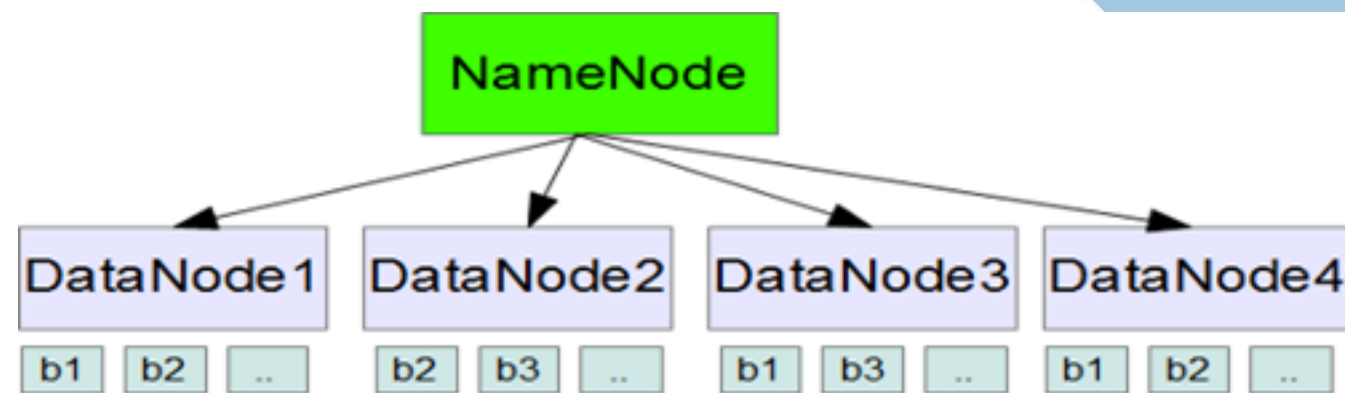
COMPONENTES HDFS

- Está basado en dos tipos de nodos:
 - NameNode
 - DataNode
- Para una instalación mínima de Hadoop, se requiere un daemon NameNode y un daemon DataNode en al menos una máquina.
- Se trata de una arquitectura master/esclavo, en la que el master (NameNode) administra el sistema de archivos y regula el acceso por parte de los clientes.
- También determina el mapeo de los bloques y las fallas de los DataNodes.



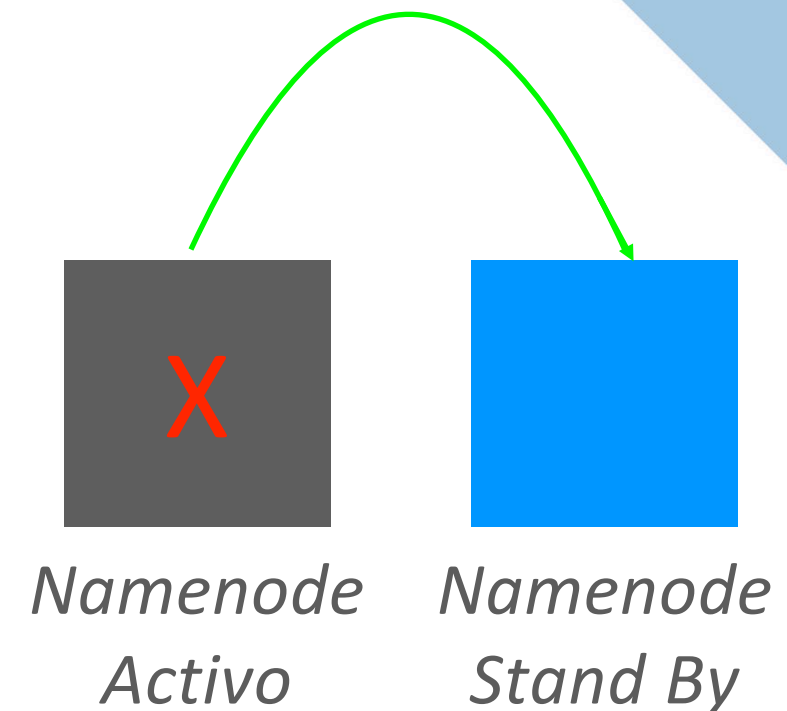
COMPONENTES HDFS

- Los esclavos (DataNodes) son responsables de cumplir los requerimientos de lectura y escritura del sistema de archivos de los clientes,
- El NameNode administra la creación, borrado y replicado de bloques.
- Como parte del almacenado, los bloques de datos son replicados después de ser escritos al nodo asignado.
- Dependiendo de la cantidad de nodos, el NameNode buscará escribir replicas de los nodos en diferentes racks.
- Si existe un solo rack, las réplicas son escritas en otros servidores en el mismo rack.
- Una vez que el proceso de escritura ha concluido, el DataNode informa al NameNode.
- El NameNode espera que se complete la operación, de no hacerlo, cancela el proceso.



COMPONENTES HDFS

- La lectura ocurre en un proceso similar, el cliente solicita un archivo del NameNode, el cual retorna los DataNodes de cuales leer.
- Posteriormente el cliente accede a lo datos directamente de los DataNodes.
- Mientras se realiza la transferencia, el NameNode monitorea a los DataNodes al escuchar sus latidos.
- En caso de no escucharlos indica una potencial falla. En su caso, ruteará el nodo fallado a otro y comenzará nuevamente el proceso de réplica.



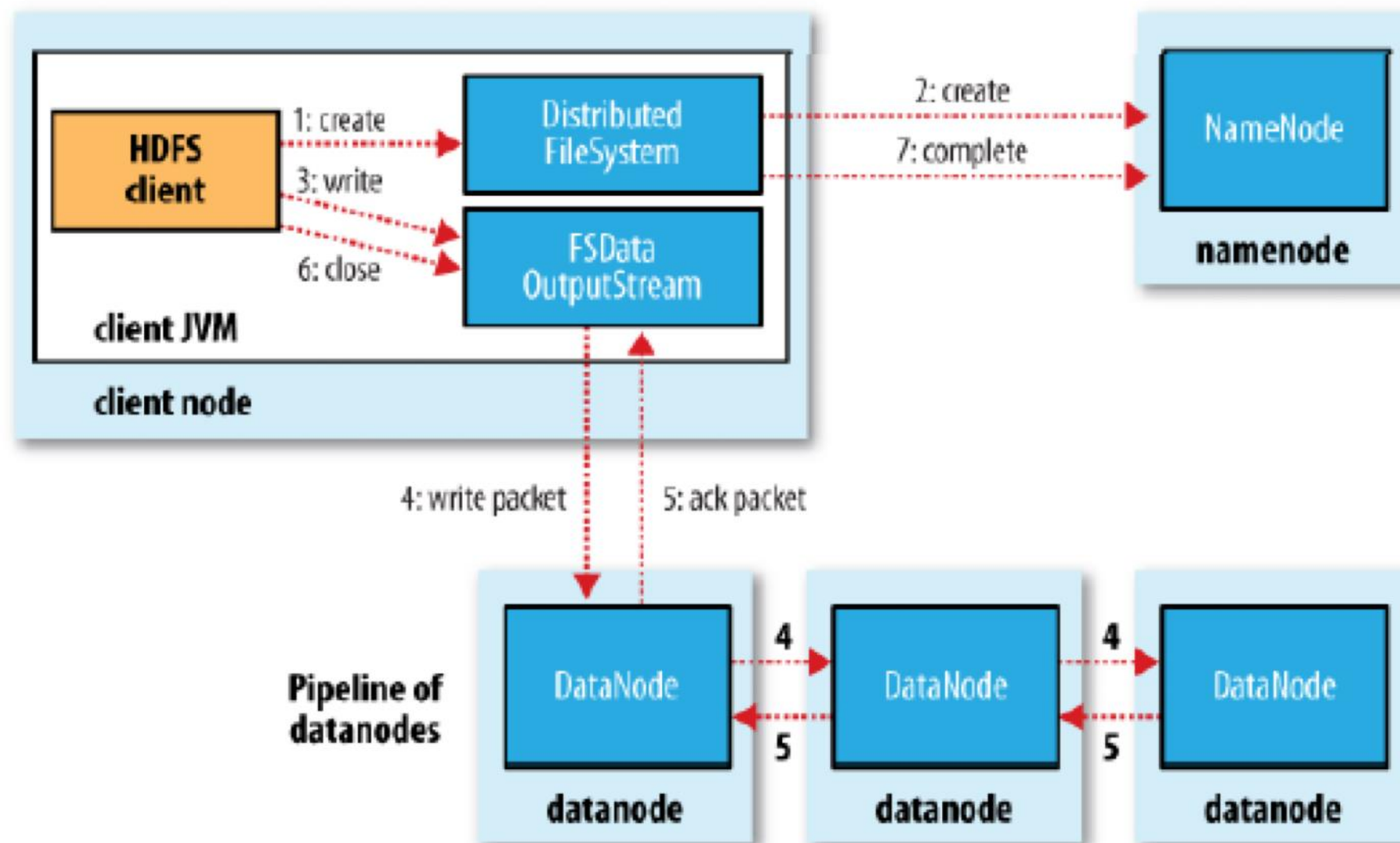
COMPONENTES HDFS

- El NameNode guarda todos los metadatos en memoria.
- Cada DataNode provee un reporte de bloque al NameNode.
- Los reportes de bloque son enviados cada 10 latidos pero se puede cambiar esta configuración.



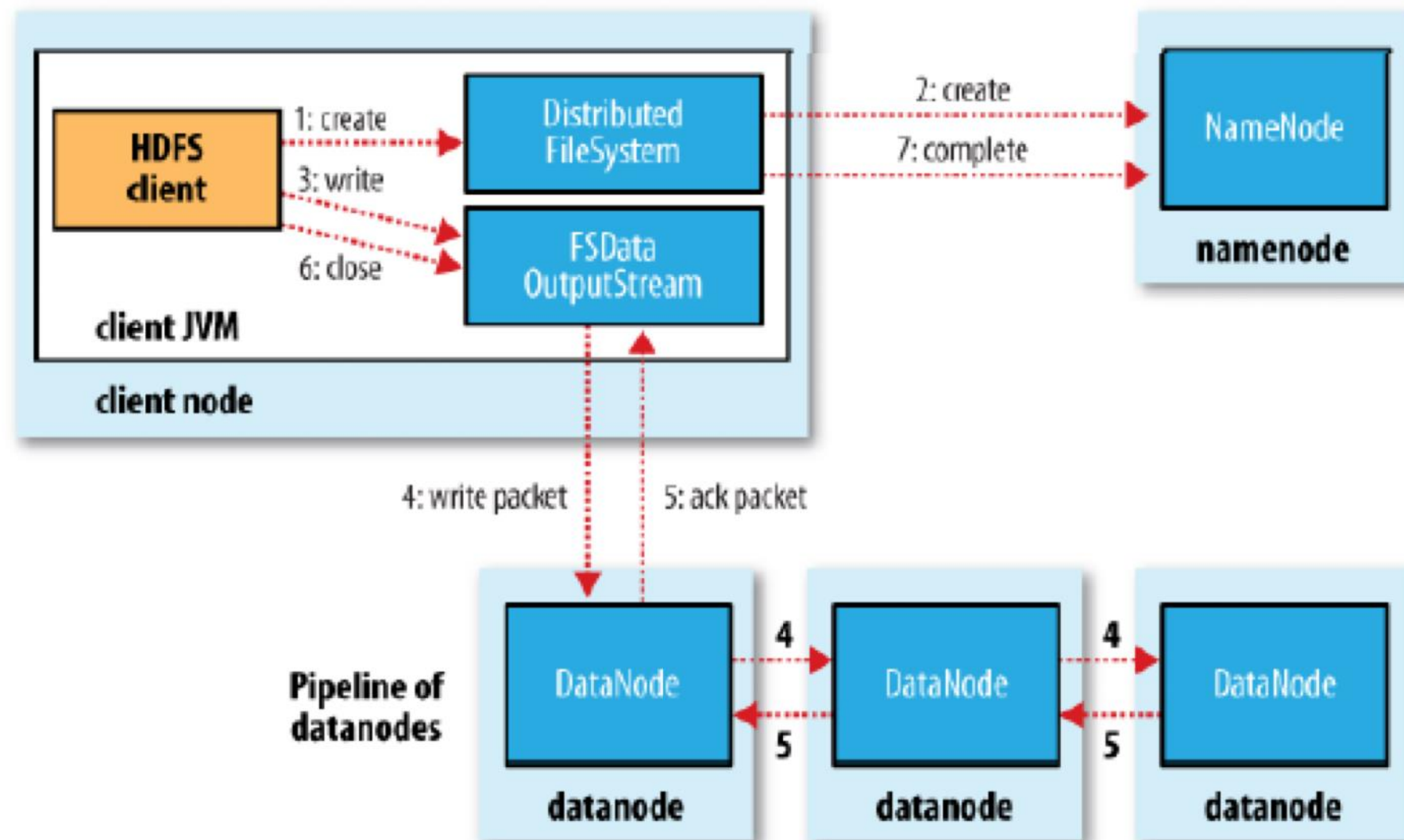
COMPONENTES HDFS

- Los clientes crean el archivo al llamar el método create()
- NameNode valida y procesa la petición
- Divide el archivo en paquetes (DataQueue)
- DataStreamer solicita al NameNode por bloques / node mapping & pipelines son creados en los nodos.



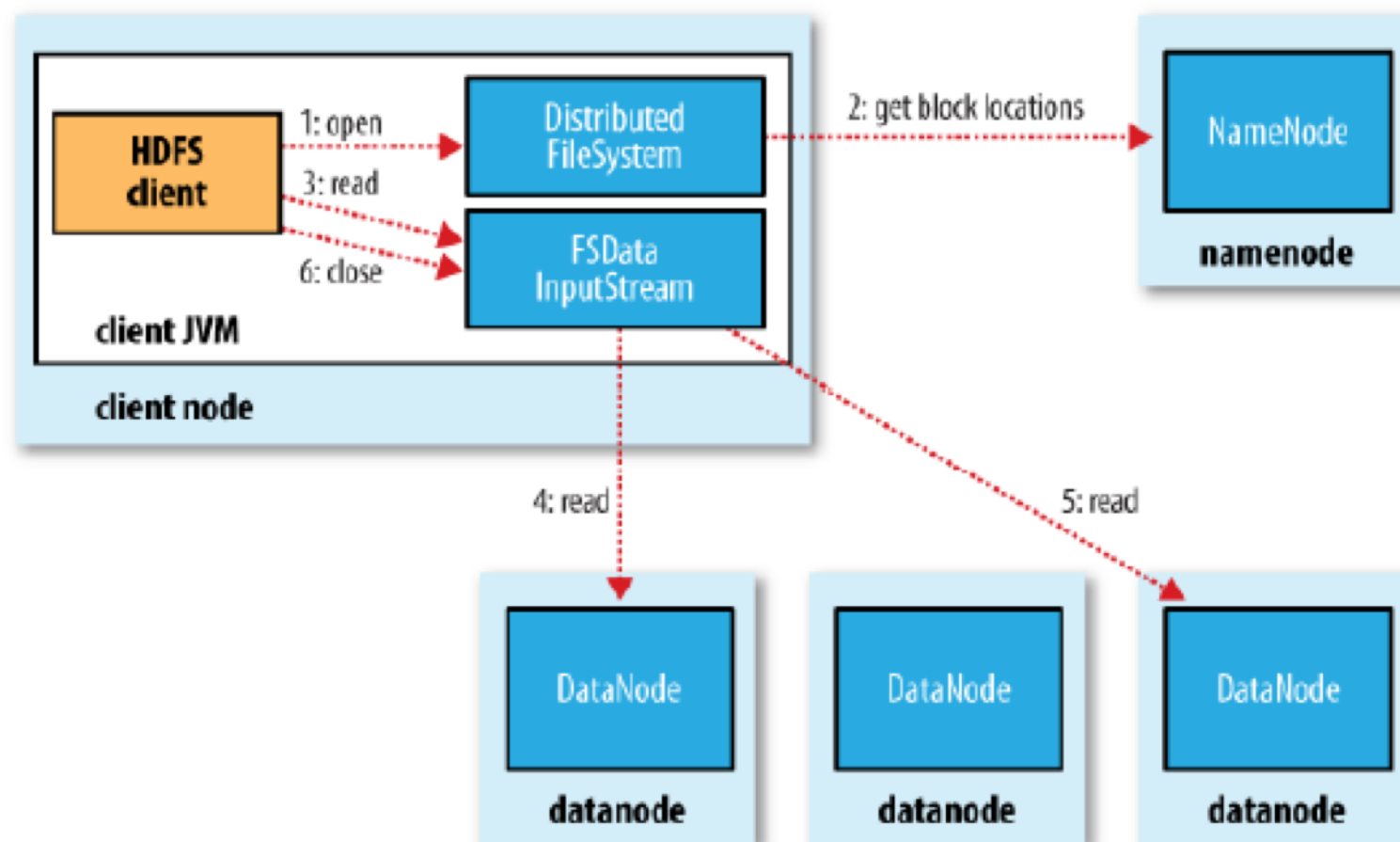
COMPONENTES HDFS

- DataStreamer envía en streams los paquetes al primer DataNode
- DataNode reenvía los paquetes copiados al siguiente DataNode en el pipeline
- DFSOutputStream también mantiene el paquete queue y remueve los paquetes después de tener conocimiento de los DataNodes
- El cliente llama close() sobre el stream



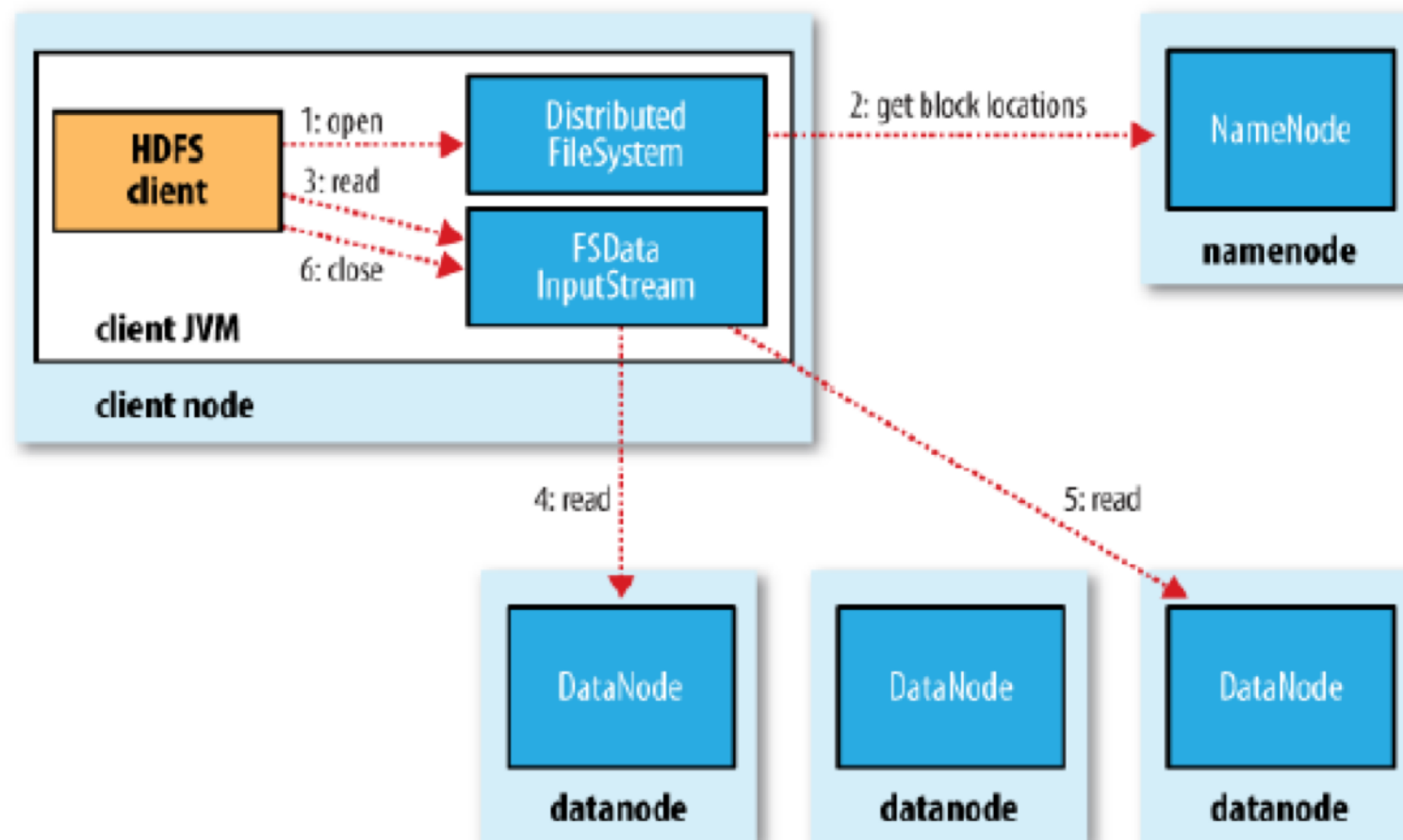
COMPONENTES HDFS

- Cliente llama open() sobre el objeto del FileSystem.
- DistributedFileSystem llama al NameNode para determinar la localización de los bloques.
- NameNode valida la petición y por cada bloque devuelve la lista de los DataNodes.
- DistributedFileSystem devuelve un stream insumo que soporta el archivo buscado por el cliente.



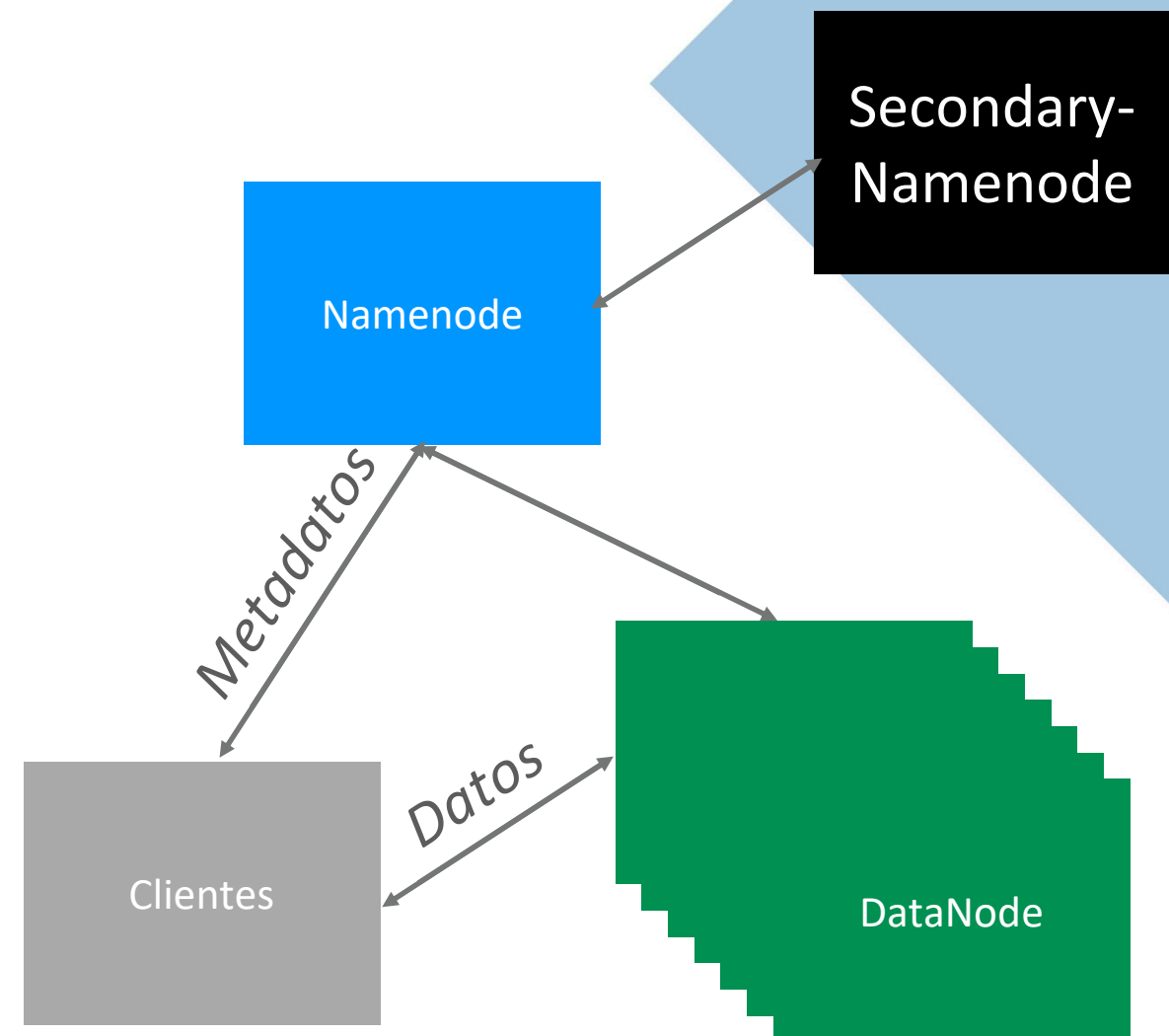
COMPONENTES HDFS

- Cliente llama read() sobre stream.
- Cuando se alcanza el final del bloque, DFSInputStream cerrará la conexión al DataNode, después encontrará el DataNode para el siguiente bloque.
- Client llamará close() sobre el stream.



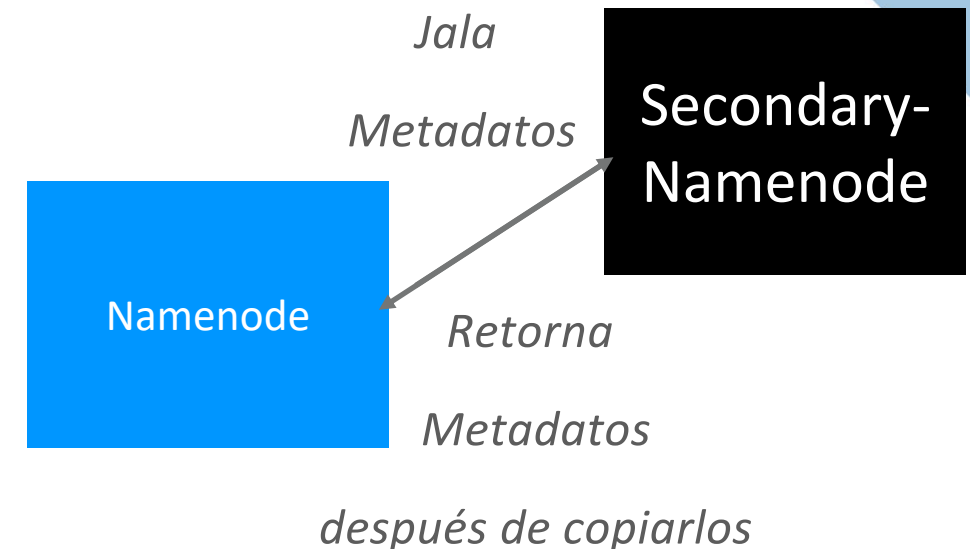
COMPONENTES HDFS

- En muchas implementaciones es recomendable crear un SecondaryNameNode, también llamado CheckPointNode.
- El objetivo de este es ejecutar checkpoints de forma periódica para evaluar el estatus del NameNode.
- Tiene dos discos de archivos que rastrean cambios en los metadatos en los siguientes aspectos:
 - Una imagen del estado del sistema de archivos al inicio.
 - Una serie de modificaciones hechas al sistema de archivos después de iniciado.
- La ubicación de estos archivos está dada por la propiedad `dfs.namenode.name.dir` en el archivo `hfs-site.xml`.



COMPONENTES HDFS

- El SecondaryNameNode descarga de forma periódica fsimage, edita archivos, los une en un nuevo fsimage y carga el nuevo fsimage al NameNode.
- Al reiniciar el NameNode, el archivo fsimage está razonablemente actualizado y requiere sólo aplicar los cambios a los logs ocurridos después del último checkpoint.

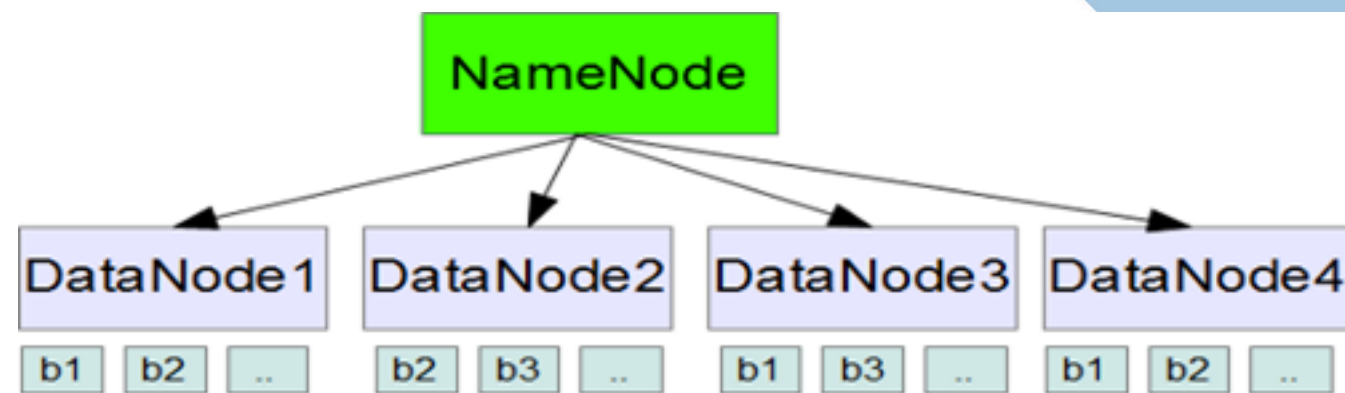


COMPONENTES HDFS

- En síntesis, los varios roles de HDFS son:
 - HDFS utiliza el modelo master-esclavo diseñado para grandes lectura-streaming.
 - El NameNode es el servidor de Metadatos.
 - HDFS provee un solo namespace administrado por NameNode.
 - Los datos son almacenados de forma redundante en los DataNodes. No hay datos en el NameNode.
 - El SecondaryNameNode hace checkpoints del NameNode, pero no es un nodo para recuperación ante fallos.

COMPONENTES HDFS

- HDFS replica de Bloques.
- Al escribir HDFS un archivo es replicado en el cluster.
- El número de réplicas lo encuentran en el archivo `hdfs-site.xml` en el valor `dfs.replication`.
- Para clusters que contienen más de 8 nodos se recomienda un valor de 3, 8 o menos y más de 1, se recomienda 2.
- El tamaño de bloque estándar es 128MB.



```
<property>  
  <name>dfs.replication</name>  
  <value>3</value>  
</property>  
  
<property>  
  <name>dfs.replication.max</name>  
  <value>50</value>  
</property>
```

COMPONENTES HDFS

- HDFS Safe Mode
- Al iniciar el NameNode entra en un estado safe mode tal que los bloques no pueden ser replicados o borrados.
- Esto tiene el objetivo de ejecutar los siguientes dos procesos:
 - Se reconstruye el estado previo del sistema de archivos al cargar el archivo fsimage en memoria y registrando el log edit.
 - El mapeo entre bloques y nodos de datos es creado esperando que los DataNodes se registren y teniendo al menos una copia de los datos disponible. El proceso de registro puede continuar posterior a la salida del Safe Mode.

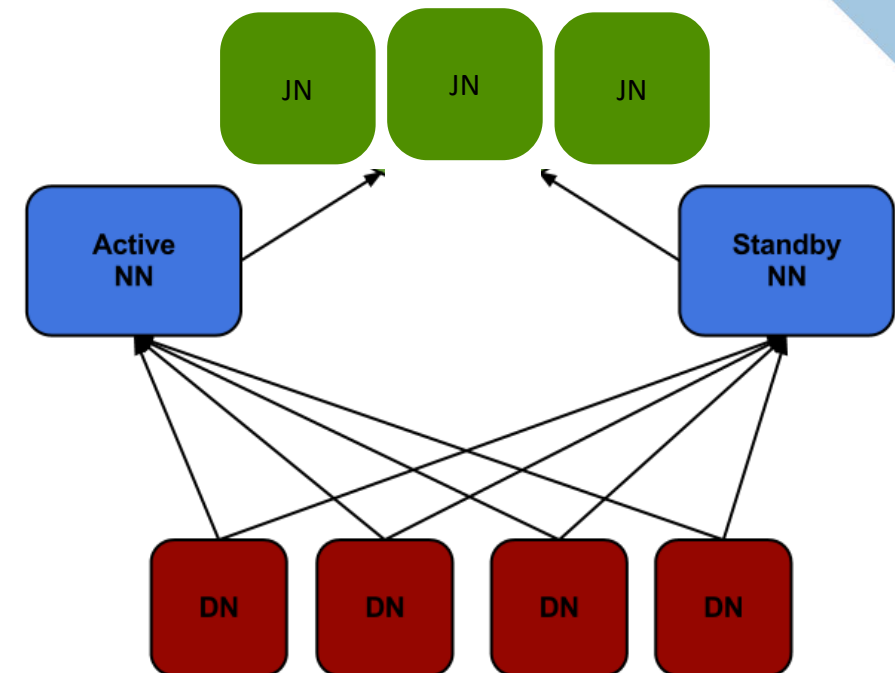
COMPONENTES HDFS

- Rack Awareness
- Está dirigido a la localidad de los datos.
- Un cluster de Hadoop típico tiene tres niveles de localidad:
 - Los datos residen en la máquina local (best).
 - Los datos residen en el mismo rack (better).
 - Los datos residen en diferente rack (good).
- YARN buscará asignarlo en este orden al asignar los contenedores que trabajen como mappers.
- Además, el NameNode tratará ubicar las réplicas de los bloques de los datos en múltiples racks para mejorar la tolerancia a fallos.
- HDFS puede hacer rack-aware al utilizar un script de usuario que habilite al nodo master mapear la topología de la red del cluster.



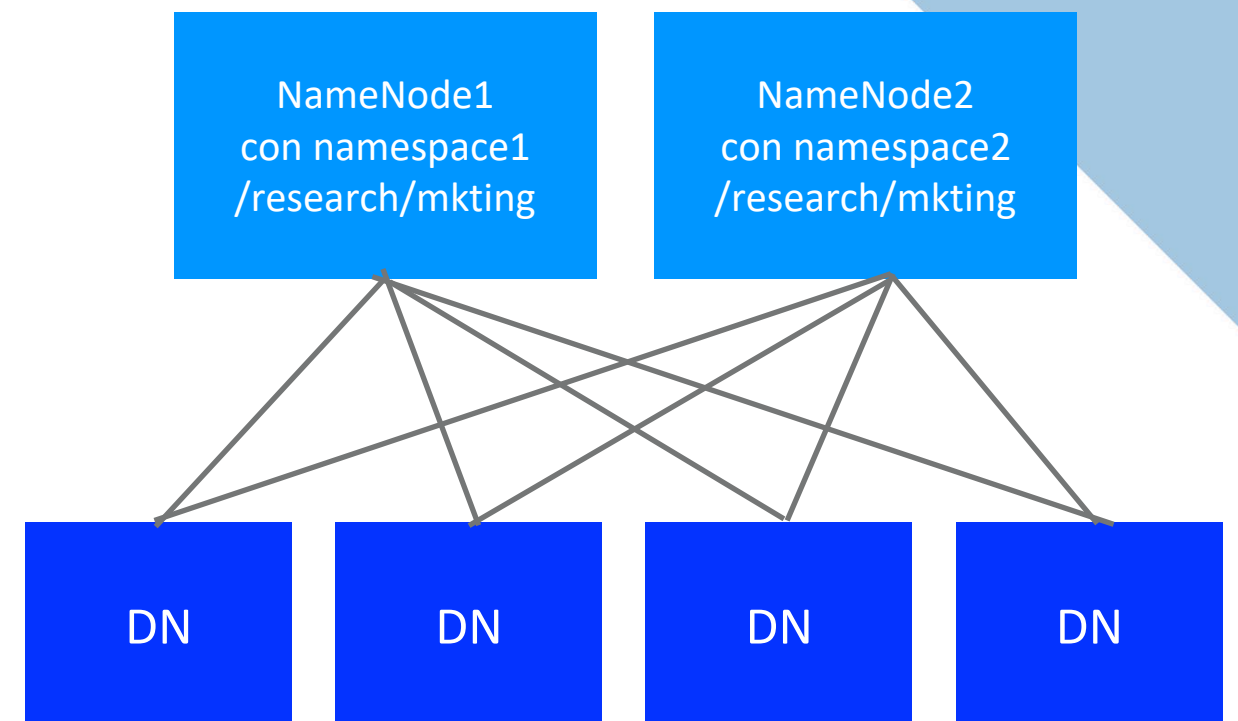
COMPONENTES HDFS

- Alta disponibilidad NameNode
- En las primeras implementaciones, el NameNode era un punto de falla del cluster entero.
- La solución fue crear un nodo de Alta Disponibilidad.
- Para garantizar que se preserve el estado del sistema de archivos, tanto el NameNode Activo como el StandBy reciben los bloques de reportes.
- Al menos se requieren tres daemons JournalNodes dado que las modificaciones a los logs se enviarán a los JournalNodes.
- El Nodo StandBy lee de forma continua las ediciones al JournalNodes para asegurar que el namespace este sincronizado con el nodo Activo.
- En caso de falla, el StandBy leerá lo existente en los JN previo a tomar el control en estado Activo.



COMPONENTES HDFS

- Federación NameNode HDFS
- Permite desvincular el namespace a los recursos de un sólo NameNode.
- Los beneficios son:
 - Escalabilidad de namespace.
 - Mejor desempeño.
 - Aislado del sistema.
- Los NameNodes en una federación no se comunican entre si, simplemente administran su espacio de namespace.



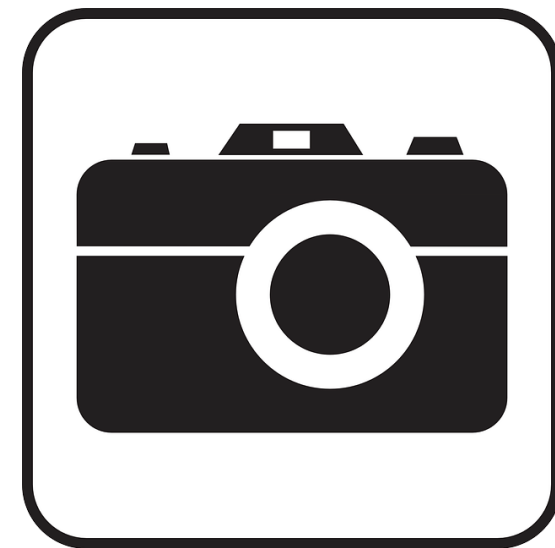
COMPONENTES HDFS

- CheckPoints y Respaldos
- HDFS BackupNode mantiene una copia actualizada del sistema de archivos namespace tanto en memoria como en disco.
- A diferencia del CheckpointNode, no debe descargar los archivos fsimage y edits del NameNode activo porque ya lo tiene en memoria.
- NameNode soporta un BackupNode a la vez.
- No se registran CheckpointNodes si existe un BackupNode.



COMPONENTES HDFS

- HDFS Snapshot
- Son similares a los backups, pero son creados por los administradores utilizando el comando
- `hdfs dfs -snapshot`
- Tienen los siguientes atributos:
 - pueden ser tomados de un sub-tree del sistema de archivos o completo.
 - se pueden utilizar para respaldos, protección ante errores de usuarios, recuperación de desastres.
 - Su creación es instantánea.
 - Los bloques de los DataNodes no son copiados. sólo copia la lista de bloques y su tamaño.
 - No afecta las operaciones regulares del HDFS.



COMPONENTES HDFS

- HDFS NFS Gateway
- Soporta NFSv3 y permite al HDFS ser montado como parte del sistema de archivos local del cliente.
- Lo anterior permite:
 - Cargar-Descargar archivos del-al HDFS del-al sistema de archivos local.
 - Se puede enviar datos directo al HDFS a través del punto donde se montó. Se puede hacer append, pero la escritura aleatoria no es soportada.



COMANDOS HDFS

- A continuación se presenta una lista de comandos HDFS para facilitar su comprensión.
- <https://data-flair.training/blogs/top-hadoop-hdfs-commands-tutorial/>

COMANDOS HDFS

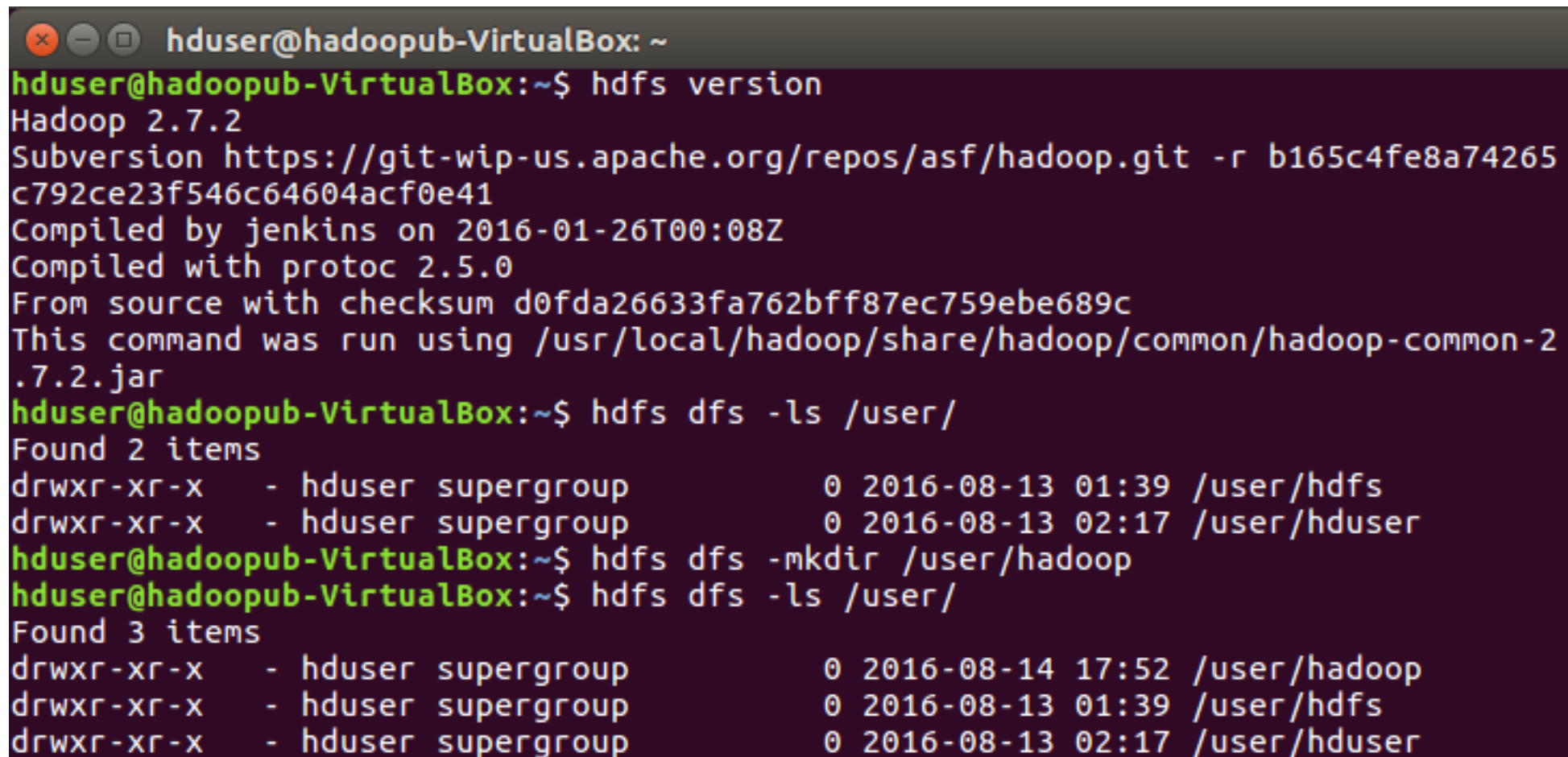
- Revisa los directorios activos

```
hdfs dfs -ls /
```

- A continuación haremos uso de Hadoop mediante comandos muy similares a los de Unix.

```
hdfs dfs -mkdir /user/hadoop
```

```
hdfs dfs -ls /
```



```
hduser@hadoopub-VirtualBox: ~
hduser@hadoopub-VirtualBox:~$ hdfs version
Hadoop 2.7.2
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r b165c4fe8a74265c792ce23f546c64604acf0e41
Compiled by jenkins on 2016-01-26T00:08Z
Compiled with protoc 2.5.0
From source with checksum d0fda26633fa762bffa87ec759ebe689c
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-2.7.2.jar
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /user/
Found 2 items
drwxr-xr-x - hduser supergroup          0 2016-08-13 01:39 /user/hdfs
drwxr-xr-x - hduser supergroup          0 2016-08-13 02:17 /user/hduser
hduser@hadoopub-VirtualBox:~$ hdfs dfs -mkdir /user/hadoop
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /user/
Found 3 items
drwxr-xr-x - hduser supergroup          0 2016-08-14 17:52 /user/hadoop
drwxr-xr-x - hduser supergroup          0 2016-08-13 01:39 /user/hdfs
drwxr-xr-x - hduser supergroup          0 2016-08-13 02:17 /user/hduser
```

COMANDOS HDFS

- En esta sección se crea un archivo de texto y se carga al hdfs.

```
cd /home/hduser
```

```
ls
```

```
echo "Esto es una prueba" >> text.txt
```

```
hdfs dfs -copyFromLocal text.txt /user/hadoop
```

```
hdfs dfs -ls /user/hadoop
```

```
hduser@hadoopub-VirtualBox:~$ cd /home/hduser
hduser@hadoopub-VirtualBox:~$ ls
examples.desktop  hadoop-2.7.2  hadoop-2.7.2.tar.gz
hduser@hadoopub-VirtualBox:~$ echo "Esto es una prueba" >> text.txt
hduser@hadoopub-VirtualBox:~$ ls
examples.desktop  hadoop-2.7.2  hadoop-2.7.2.tar.gz  text.txt
hduser@hadoopub-VirtualBox:~$ hdfs dfs -copyFromLocal text.txt /user/hadoop
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /user/hadoop
Found 1 items
-rw-r--r--  1 hduser supergroup      19 2016-08-14 18:12 /user/hadoop/text.t
xt
```


COMANDOS HDFS

- También podemos leer el archivo cargado desde hdfs.

ls

rm text.txt

cat text.txt

hdfs dfs -ls /

hdfs dfs -cat /user/hadoop/text.txt

hdfs dfs -copyToLocal /user/hadoop/text.txt

```
hduser@hadoopub-VirtualBox:~$ ls
examples.desktop  hadoop-2.7.2  hadoop-2.7.2.tar.gz  text.txt
hduser@hadoopub-VirtualBox:~$ rm text.txt
hduser@hadoopub-VirtualBox:~$ cat text.txt
cat: text.txt: No such file or directory
hduser@hadoopub-VirtualBox:~$ hdfs dfs -cat /user/hadoop/text.txt
Esto es una prueba
hduser@hadoopub-VirtualBox:~$ hdfs dfs -copyToLocal /user/hadoop/text.txt
hduser@hadoopub-VirtualBox:~$
```

COMANDOS HDFS

- Se puede remover archivos o directorios completos de hfs.

hdfs dfs -ls /user/

hdfs dfs -rm -r /user/hdfs

hdfs dfs -ls /user

```
hduser@hadoopub-VirtualBox: ~  
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /user/  
Found 3 items  
drwxr-xr-x  - hduser supergroup          0 2016-08-14 18:12 /user/hadoop  
drwxr-xr-x  - hduser supergroup          0 2016-08-13 01:39 /user/hdfs  
drwxr-xr-x  - hduser supergroup          0 2016-08-13 02:17 /user/hduser  
hduser@hadoopub-VirtualBox:~$ hdfs dfs -rm -r /user/hdfs  
16/08/14 18:37:38 INFO fs.TrashPolicyDefault: Namenode trash configuration: Dele  
tion interval = 0 minutes, Emptier interval = 0 minutes.  
Deleted /user/hdfs  
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /user/  
Found 2 items  
drwxr-xr-x  - hduser supergroup          0 2016-08-14 18:12 /user/hadoop  
drwxr-xr-x  - hduser supergroup          0 2016-08-13 02:17 /user/hduser  
hduser@hadoopub-VirtualBox:~$
```

COMANDOS HDFS

- Se puede obtener un reporte del estado de hfs.
- Los administradores recibirán un reporte largo, mientras que los usuarios generales uno corto.

hdfs dfsadmin -report

```
hduser@hadoopub-VirtualBox: ~  
hduser@hadoopub-VirtualBox:~$ hdfs dfsadmin -report  
Configured Capacity: 30779944960 (28.67 GB)  
Present Capacity: 24495173632 (22.81 GB)  
DFS Remaining: 24494854144 (22.81 GB)  
DFS Used: 319488 (312 KB)  
DFS Used%: 0.00%  
Under replicated blocks: 0  
Blocks with corrupt replicas: 0  
Missing blocks: 0  
Missing blocks (with replication factor 1): 0  
  
-----  
Live datanodes (1):  
  
Name: 127.0.0.1:50010 (localhost)  
Hostname: hadoopub-VirtualBox  
Decommission Status : Normal  
Configured Capacity: 30779944960 (28.67 GB)  
DFS Used: 319488 (312 KB)  
Non DFS Used: 6284771328 (5.85 GB)  
DFS Remaining: 24494854144 (22.81 GB)  
DFS Used%: 0.00%  
DFS Remaining%: 79.58%  
Configured Cache Capacity: 0 (0 B)  
Cache Used: 0 (0 B)  
Cache Remaining: 0 (0 B)  
Cache Used%: 100.00%  
Cache Remaining%: 0.00%  
Xceivers: 1  
Last contact: Sun Aug 14 18:40:13 CDT 2016
```

HDFS EN PROGRAMAS

- Guarda en un directorio data el archivo test.txt

```
hdfs dfs -mkdir /data
```

```
hdfs dfs -ls /
```

```
hdfs dfs -cp /user/hadoop/test.txt /data/
```

```
hdfs dfs -ls /data/
```

```
hduser@hadoopub-VirtualBox: ~  
hduser@hadoopub-VirtualBox:~$ hdfs dfs -mkdir /data  
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /  
Found 4 items  
drwxr-xr-x  - hduser supergroup          0 2016-08-14 18:27 /data  
drwxr-xr-x  - hduser hadoop              0 2016-08-13 01:36 /mr-history  
drwx----- - hduser supergroup          0 2016-08-13 02:15 /tmp  
drwx----- - hduser supergroup          0 2016-08-14 17:52 /user  
hduser@hadoopub-VirtualBox:~$ hdfs dfs -cp /user/hadoop/text.txt /data/  
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /data/  
Found 1 items  
-rw-r--r--  1 hduser supergroup          19 2016-08-14 18:30 /data/text.txt  
hduser@hadoopub-VirtualBox:~$
```

HDFS EN PROGRAMAS

- Ejecuta el jar de ejemplo de acuerdo a la siguientes ruta:

hadoop jar ./usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.2.jar wordcount /data out

```
hduser@hadoopub-VirtualBox: /usr/local/hadoop/share/hadoop/mapreduce
Map output records=4
Map output bytes=35
Map output materialized bytes=49
Input split bytes=100
Combine input records=4
Combine output records=4
Reduce input groups=4
Reduce shuffle bytes=49
Reduce input records=4
Reduce output records=4
Spilled Records=8
Shuffled Maps =1
Failed Shuffles=0
Merged Map outputs=1
GC time elapsed (ms)=563
CPU time spent (ms)=3110
Physical memory (bytes) snapshot=428535808
Virtual memory (bytes) snapshot=3844739072
Total committed heap usage (bytes)=304087040

Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0

File Input Format Counters
Bytes Read=19
File Output Format Counters
Bytes Written=27
```


HDFS EN PROGRAMAS

- Revisa los resultado del proceso.

hdfs dfs -ls out

hdfs dfs -cat out/part-r-00000

```
hduser@hadoopub-VirtualBox: ~
hduser@hadoopub-VirtualBox:/usr/local/hadoop/share/hadoop/mapreduce$ cd /home/hd
user
hduser@hadoopub-VirtualBox:~$ ls
examples.desktop  hadoop-2.7.2  hadoop-2.7.2.tar.gz  text.txt
hduser@hadoopub-VirtualBox:~$ hdfs dfs -ls /out
Found 2 items
-rw-r--r--    1 hduser supergroup      0 2016-08-14 21:17 /out/_SUCCESS
-rw-r--r--    1 hduser supergroup    27 2016-08-14 21:17 /out/part-r-00000
hduser@hadoopub-VirtualBox:~$ hdfs dfs -cat /out/part-r-00000
Esto      1
es        1
prueba    1
una       1
hduser@hadoopub-VirtualBox:~$
```

HDFS EN PROGRAMAS

- Si deseas tener una lista de todos los ejemplos incluidos en la versión que está utilizando de hadoop, ejecuta el siguiente comando:

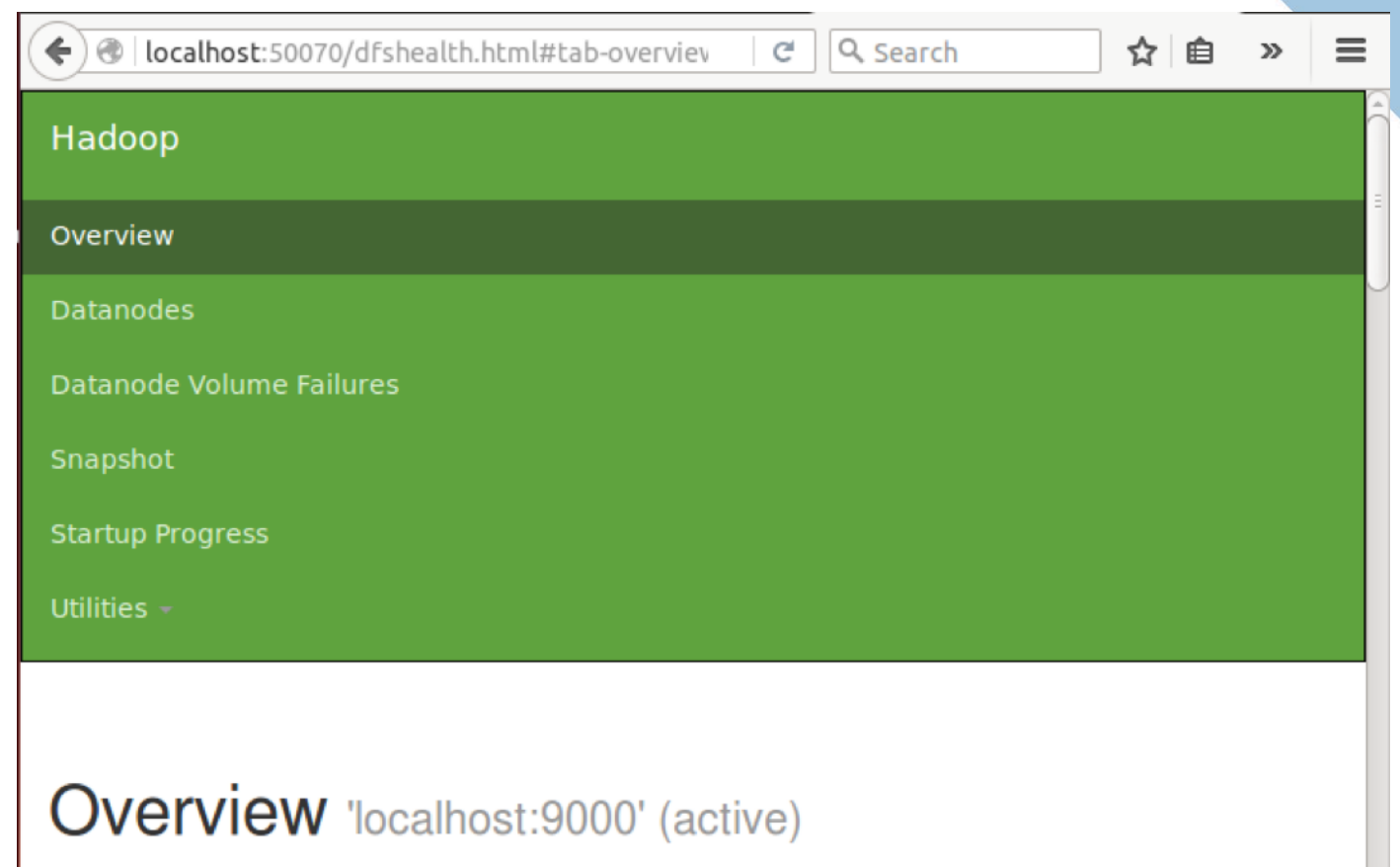
```
hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.2.jar
```

```
hduser@hadoopub-VirtualBox:~$ hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.2.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.
  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
```

WEB UI

- Para acceder a la interfaz web de usuario debes acceder desde Firefox a la siguiente dirección.

<http://localhost:50070/>



EJERCICIO

- Baja algún libro del proyecto Gutenberg y realiza un conteo de palabras similar al realizado en el ejercicio previo.
- <http://www.gutenberg.org>

BIBLIOGRAFÍA

- Turkington Garry, “Hadoop Beginner’s Guide” PACKT. 2013.
- “How to enable root login”
<http://askubuntu.com/questions/44418/how-to-enable-root-login>
- Apache Foundation. “The File System (FS)”
<https://hadoop.apache.org/docs/r2.7.2/hadoop-project-dist/hadoop-common/FileSystemShell.html>
- Eadline Douglas. “Hadoop 2 Quick-Start Guide” Addison Wesley Data & Analytics Series. 2016