TECNOLOXÍAS DE COMPUTACIÓN PARA DATOS MASIVOS

Entrega 1

**ANDRÉS CAMPOS CUIÑA**

**FECHA DE ENTREGA: 17/10/2021**

ÍNDICE

[1 Añadir al cluster un servidor de BackUp y un TimeLineServer 1](#_Toc85070904)

[1.1 Servidor de BackUp 1](#_Toc85070905)

[1.2 Servidor TimeLineServer 5](#_Toc85070906)

[2 Añadir un nuevo DataNode/NodeManager 7](#_Toc85070907)

[2.1 Creación de ficheros de nodos incluidos y excluidos 7](#_Toc85070908)

[2.2 Añadir un datanode/nodemanager 9](#_Toc85070909)

[3 Retirar un DataNode/NodeManager 12](#_Toc85070910)

[4 Rack awareness 13](#_Toc85070911)

# Añadir al cluster un servidor de BackUp y un TimeLineServer

Procederemos a añadir al cluster configurado anteriormente un servidor de BackUp y un servidor TimeLineServer. Se irán explicando los pasos llevados a cabo a continuación.

## Servidor de BackUp

En primer lugar, iniciamos el clúster y en el NameNode obtenemos una captura en la que se muestren los ficheros del directorio de metadatos del NameNode. Para esto lo primero que hacemos es desplazarnos al directorio donde se encuentran estos ficheros mediante cd /var/data/hdfs/namenode/current y después comprobamos que ficheros hay mediante el comando ls -all. El resultado es el siguiente:

Texto

Descripción generada automáticamente

Para añadir el servidor de BackUp seguimos los pasos descritos en el guión de esta entrega. En primer lugar, con el cluster funcionando, ejecutamos el comando docker container run -ti --name backupnode --network=hadoop-cluster --hostname backupnode --cpus=1 --memory=3072m --expose 50100 -p 50105:50105 hadoop-base /bin/bash para iniciar un nuevo Docker a partir de la imagen hadoop-base.

Una vez que ya se ha creado este nuevo contenedor, de nombre backupnode, accedemos a él y creamos en el directorio raíz el directorio /backup mediante el comando mkdir backup, hacemos que el propietario sea hdadmin mediante el comando chown hdadmin:hadoop /backup y por último creamos dentro de este directorio la carpeta dfs/name (hacemos que el usuario hdadmin sea el propietario de estas carpetas también).

Después, cambiamos al usuario hdadmin (mediante su – hdadmin) y después modificamos el fichero core-site.xml para añadir las propiedades que se nos especifican en el enunciado:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

También añadimos las propiedades necesarias al fichero **hdfs-site.xml**:

Texto

Descripción generada automáticamente

Ahora iniciamos el servidor de backup ejecutando el comando hdfs namenode -backup. Unos ejemplos de los mensajes que nos da el servicio de backup, destacando aquellos en los que se ve como se hace el checkpoint, al iniciarse son los siguientes:

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Como se puede ver en la última línea en los mensajes del servicio de backup, se nos dice que un checkpoint ha sido correctamente completado en 6 segundos.

A continuación, podemos una captura de pantalla de la interfaz web del nodo de backup:

Interfaz de usuario gráfica, Texto, Correo electrónico

Descripción generada automáticamente

Finalmente, compararemos el contenido del directorio del backup con el directorio con los metadatos del NameNode, antes y después de que el servicio de backup se haya completado. En primer lugar, vemos el contenido del directorio con los metadatos del NameNode después de haber ejecutado el servicio de backup (anteriormente ya se mostró la captura con los contenidos de este directorio antes de ejecutar el servicio de backup):

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

En la siguiente captura se puede ver el contenido del directorio de backup después de haber ejecutado el servicio de backup (antes de la ejecución este directorio se encontraba vacío):

Texto

Descripción generada automáticamente

Para poder entender los contenidos de estos directorios es mejor primero dejar claro que los ficheros cuyo nombre empieza por ‘fsimage’ son los ficheros que contienen la estructura de directorios completa (namespace) de HDFS con información sobre la localización de cada fichero. Los ficheros cuyo nombre empieza por ‘edits’ tienen los cambios realizados al HDFS desde que la última FsImage fue creada. Por lo que con la última FsImage y los ficheros EditLogs podemos restaurar el estado del HDFS en caso de fallo.

Entonces, en las capturas mostrada anteriormente podemos ver como al ejecutar el servicio de backup se añade un fichero FsImage en el directorio del NameNode y este mismo fichero se almacena en el directorio de backup, además de los ficheros EditLogs que están presentes en ambos directorios. En el directorio del NameNode se almacenan dos ficheros FsImage mientras que en el directorio del backup sólo se almacena el que fue creado después de ejecutar el servicio de backup.

## Servidor TimeLineServer

En primer lugar, accedemos al NameNode y como usuario hdadmin detenemos el servicio ResourceManager mediante el comando yarn –daemon stop resourcemanager. Ahora editamos el fichero yarn-site.xml para añadir las siguientes propiedades:

Captura de pantalla de un celular

Descripción generada automáticamente

Reiniciamos el servicio ResourceManager mediante el comando yarn –daemon start resourcemanager. Utilizamos el comando docker container run -ti --name timelineserver --network=hadoop-cluster --hostname timelineserver --cpus=1 --memory=3072m --expose 10200 -p 8188:8188 hadoop-base /bin/bash para iniciar un nuevo Docker en el que levantaremos el servicio TimeLineServer. En este nuevo Docker levantamos el servicio TimeLineServer mediante yarn –daemon start timelineserve (como usuario hdadmin). Desde el NameNode ejecutamos la aplicación del cálculo de pi con YARN (como usuario hdadmin).De esta manera, el servidor web del TimeLineServer recoge la información de la ejecución de esta tarea, como se puede ver en la siguiente captura:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

# Añadir un nuevo DataNode/NodeManager

## Creación de ficheros de nodos incluidos y excluidos

lo que haremos es crear los ficheros de incluidos y de excluidos. Para ello primero detenemos los demonios del NameNode y del ResourceManager. Después creamos los cuatro ficheros que s indican en el enunciado. En los ficheros **dfs.include** y **yarn.include** incluimos los cuatro datanodes.

A continuación, editamos el fichero **hdfs.site.xml** para añadir las siguientes propiedades:

Texto

Descripción generada automáticamente

En el fichero **yarn-site.xml** añadimos otras dos propiedades:

Texto

Descripción generada automáticamente

Para terminar, reiniciamos los demonios. Mediante el comando grep -l datanode ./\* obtenemos los ficheros en el directorio de logs en los que aparezcan nombrados los datanodes. El resultado de este comando es el siguiente:

Texto

Descripción generada automáticamente

Por lo tanto, consultando en estos dos ficheros podemos ver como se han incluido al HDFS y al YARN los nodos datanode{1..4}:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

## Añadir un datanode/nodemanager

Después de haber añadido el datanode5 al fichero **dfs.include** ejecutamos el comando hdfs dfsadmin -report para comprobar como el nuevo contenedor no se ha añadido al HDFS. Como se muestra en la siguiente captura solo hay datanodes en activo:

Texto

Descripción generada automáticamente

Sin embargo, si ejecutamos el comando yarn node -list vemos como sí que se ha añadido al YARN:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Tras añadir el datanode5 al fichero **dfs.include** y actualizar el NameNode vemos como ahora sí que se ha añadido al HDFS, tal y como se muestra en la siguiente captura donde podemos ver la información del datanode5 añadido:

Texto

Descripción generada automáticamente

Finalmente, forzamos el balanceo del clúster ejecutando el comando hdfs balancer. Antes de ejecutar el balanceo el nodo datanode5 tiene 25 bloques:

Tabla

Descripción generada automáticamente

La salida el balanceador de carga es la siguiente:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Como se puede ver se han movido 0 bytes de datos. Supongo que al estar todos los nodos muy poco llenos el balanceador considera que no es necesario enviar datos al datanode5. Si los nodos datanode{1..4} estuviesen muy llenos, sí que habrían un desbalanceo considerable en el clúster y esta operación hubiese movido algunos datos al datanode5.

# Retirar un DataNode/NodeManager

Como podemos ver, tras incluir el nodo datanode4 en los ficheros **dfs.exclude** y **yarn.exclude** y actualizar el NameNode y el ResourceManager podemos ver como el nodo aparece como decomisionado en la interfaz web del HDFS:

Tabla

Descripción generada automáticamente

También podemos ver como en la interfaz web del YARN ya no aparece entre los nodos disponibles:

Tabla

Descripción generada automáticamente

# Rack awareness

Para terminar, llevamos a cabo los pasos indicados en el enunciado. El contenido del fichero **topology.data** es el siguiente:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

La propiedad mediante la cual indicamos el path al script creado:

Texto

Descripción generada automáticamente

Y por último, la salida del comando hdfs dfsadmin -printTopology en la que se ve que la distribución por racks se llevó a cabo de manera correcta:

Texto

Descripción generada automáticamente