**Ejercicios Flume**

**1. Configuración básica de Flume**

* La configuración de un agente Flume está almacenada en un fichero. A continuación, se detalla el contenido que ha de tener dicho fichero. Para este ejercicio vamos a definir:

Un agente: a1

Que escucha por un puerto: 44444

Un channel que almacena los datos en memoria

Un sink que muestra datos por consola

* La máquina virtual con la que estamos trabajando no tiene Telnet instalado, por lo que lo instalaremos. Para ello ejecutar los siguientes pasos:

yum install telnet telnet-server –y

sudo chmod 777 /etc/xinetd.d/telnet

* Editamos el archivo anterior y actualizamos la variable disable=no

vi /etc/xinetd.d/telnet

disable=no

* comprobamos que está correcto

cat /etc/xinetd.d/telnet

* Reiniciamos el servicio

sudo service xinetd start

* Hacemos que el servicio se arranque automáticamente

sudo chkconfig telnet on

sudo chkconfig xinetd on

* En el directorio /etc/flume-ng están las carpetas que contienen las plantillas que hay que configurar para realizar una importación de datos con flume.
* Para este ejercicio vamos a hacer la prueba de escribir por consola lo que escribamos a través de Telnet en un Shell. Para ello creamos un fichero llamado “example.conf” y lo guardamos en “/home/cloudera”. El contenido de este fichero es
* Abrimos un Shell. En la primera de ellas ejecutamos el siguiente comando, que arranca el agente flume.

flume-ng agent --conf conf --conf-file /home/cloudera/example.conf --name a1 -Dflume.root.logger=INFO,console

* Abrimos otro shell donde ejecutamos

telnet localhost 44444

* Ahora probamos a escribir algo en este Segundo shell, donde hemos ejecutado el telnet, y vemos cómo se envía al primer shell

**2. Importar datos de un spool-dir**

La idea de este ejercicio es ir dejando ficheros en un directorio (spool-dir) y ver como flume los va consumiendo.

* Creamos el directorio spool y le damos permisos

sudo mkdir -p /var/log/apache/flumeSpool

sudo chmod 777 /var/log/apache/flumeSpool

* Tendríamos que crear también los directorios checkpoint y datadir. Si no lo hacemos, flume lo crea por nosotros. Para poder utilizarlo le damos permisos a dicho directorio, ya que sabemos dónde se va a montar. A continuación, les damos permisos

sudo mkdir -p /mnt/flume/checkpoint

sudo mkdir -p /mnt/flume/data

sudo chmod 777 /mnt/flume/checkpoint

sudo chmod 777 /mnt/flume/data

* Creamos un fichero de configuración en la misma ruta que en el ejemplo anterior, y modificamos la configuración del source, cambArrancamos flume en un shell

flume-ng agent --conf conf --conf-file /home/cloudera/example2.conf --name a1 -Dflume.root.logger=DEBUG,console -Dorg.apache.flume.log.printconfig=true -Dorg.apache.flume.log.rawdata=true

* Para comprobar que funciona, abrimos una nueva Shell, nos posicionamos en la ruta donde hemos definido el spool-dir y creamos un fichero con el editor vi (recomendable) o con el explorador de archivos de Linux.
* Prestar atención al Shell donde tenemos flume corriendo y ver cómo se envían y muestran los ficheros por consola.

En este caso, yo personalmente lo hice con el explorador, y se reflejaba la creación del txt en consola.

**3. Importar datos desde un spool-dir a HDFS**

* Creamos el directorio en HDFS donde vamos a dejar los datos importados desde el spool-dir a través del channel de flume

hadoop fs -mkdir /flume

hadoop fs -mkdir /flume/events

* Creamos un nuevo fichero de configuración, example3.conf, igual que el del ejemplo anterior, pero sustituyendo la descripción del sink por lo siguiente

a1.sinks.k1.type = hdfs

a1.sinks.k1.hdfs.path=/flume/events

* Corremos el agente flume

flume-ng agent --conf conf --conf-file /home/cloudera/example3.conf --name a1 -Dflume.root.logger=DEBUG,console -Dorg.apache.flume.log.printconfig=true -Dorg.apache.flume.log.rawdata=true

* Nos posicionamos en el directorio spool y creamos un fichero con algo escrito. Después accedemos a la carpeta HDFS donde se supone que debe estar y vemos si está. Tarda un poco.
* Prestad atención al nombre de la carpeta
* Para mejorar un poco la info que nos devuelve Flume, añadimos la siguiente configuración de hdfs.

a1.sinks.k1.hdfs.path = /flume/events/%y-%m-%d/%H%M/%S

a1.sinks.k1.hdfs.filePrefix = events-

a1.sinks.k1.hdfs.round = true

a1.sinks.k1.hdfs.roundValue = 10

a1.sinks.k1.hdfs.roundUnit = minute

* Observad cómo cambia la estructura de carpetas donde se almacenan los datos en el sink. Creamos un nuevo fichero en el spool y vamos a HDFS para ver cómo se ha importado.

(hadoop fs -cat /flume/events/\*

hadoop fs -rm /flume/events/\*)

* Si habéis abierto uno de los ficheros de datos importados, os habréis dado cuenta de que el contenido del fichero que enviáis tiene caracteres extraños. Esto es porque por defecto flume escribe datos serializados (….BytesWritable). Si recordáis del primer día de clase, una de las propiedades de Hadoop es que serializa los datos con los que trabaja (interfaz Writable). Existe una forma de solucionar esto, que es lo que tenéis que hacer en este punto. Acceded a la web de flume y buscad la propiedad que hace que se muestren los datos en formato Texto:

a1.sinks.k1.hdfs.writeFormat = Text

a1.sinks.k1.hdfs.fileType = DataStream