Apuntes

Apache Flume es una herramienta que sirve para recopilar y transferir datos en streaming.

Es realmente útil ya que es fácil de utilizar, tanto que no es necesario una base de programación para saber utilizarlo.

Se compone de 3 partes:

Source: origen de los datos

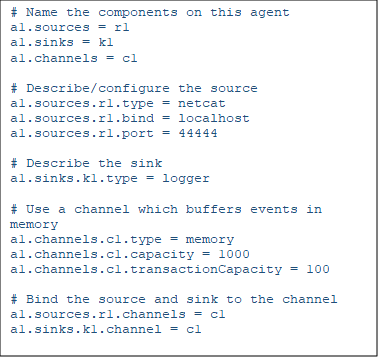
Sink: destino de los datos

Channel: Es el buffer de almacenamiento intermedio que conecta al source y al sink

Es posible crear varios agentes Flume. Para ello, se necesita que el sink del agente previo y el source del actual tienen que ser de tipo AVRO (sistema de serialización de datos), con el sink apuntando al hostname y puerto del source.

Ejercicios

1. La configuración de un agente Flume está almacenada en un fichero. A continuación, se detalla el contenido que ha de tener dicho fichero.   
   La máquina virtual con la que estamos trabajando no tiene Telnet instalado, por lo que utilizaremos la herramienta Netcat.
2. Para este ejercicio vamos a hacer la prueba de escribir por consola lo que escribamos a través de Netcat en un Shell. Para ello creamos un fichero llamado “example.conf” y lo guardamos en “/home/cloudera”. El contenido de este fichero es



1. Abrimos un Shell. En la primer de ellas ejecutamos el siguiente comando, que arranca el agente flume.
   1. flume-ng agent --conf conf --conf-file /home/cloudera/example4.conf --name a1 -Dflume.root.logger=INFO,console
2. Abrimos otro shell donde ejecutamos
   1. Nc localhost 44444
3. 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**

Para el segundo ejercicio, utilizaremos spooldir, que es un source predefinido por Flume que consiste en configurar el origen del agente Flume a un directorio en específico, y se encargará de ir leyendo el contenido de todos los ficheros dentro del directorio.

1. Creamos el directorio spool y le damos permisos
   1. sudo mkdir -p /var/log/apache/flumeSpool
   2. sudo chmod 777 /var/log/apache/flumeSpool
2. 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
   1. sudo mkdir -p /mnt/flume/checkpoint
   2. sudo mkdir -p /mnt/flume/data
   3. sudo chmod 777 /mnt/flume/checkpoint
   4. sudo chmod 777 /mnt/flume/data
3. Creamos un fichero de configuración en la misma ruta que en el ejemplo anterior, y modificamos la configuración del source, cambiándola por esta
   1. a1.sources.r1.type = spooldir
   2. a1.sources.r1.spoolDir = /var/log/apache/flumeSpool
   3. a1.sources.r1.fileHeader = true
4. Arrancamos flume en un shell
   1. 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
5. 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.
   1. Cd /var/log/apache/flumeSpool
   2. Sudo vi fichero.txt
6. Prestar atención al Shell donde tenemos flume corriendo y ver cómo se envían y muestran los ficheros por consola.

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

En el último ejercicio, modificamos el destino al sistema de ficheros Hadoop.

1. Creamos el directorio en HDFS donde vamos a dejar los datos importados desde el spool-dir a través del channel de flume
   1. hadoop fs -mkdir /flume
   2. hadoop fs -mkdir /flume/events
2. Creamos un nuevo fichero de configuración, example3.conf, igual que el del ejemplo anterior, pero sustituyendo la descripción del sink por el tipo HDFS y el path que acabamos de crear en el paso anterior
   1. a1.sinks.k1.type = hdfs
   2. a1.sinks.k1.bind = /flume/events
3. Corremos el agente flume
   1. 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
4. 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. Deteneos un momento en observar el resultado y abrid uno de los ficheros importados para ver su contenido.
5. Prestad atención al nombre de la carpeta
6. Para mejorar un poco la info que nos devuelve Flume, añadimos la siguiente configuración de hdfs.
   1. a1.sinks.k1.hdfs.path = /flume/events/%y-%m-%d
   2. a1.sinks.k1.hdfs.useLocalTimeStamp = true
   3. a1.sinks.k1.hdfs.filePrefix = events-
   4. a1.sinks.k1.hdfs.round = true
   5. a1.sinks.k1.hdfs.roundValue = 10
   6. a1.sinks.k1.hdfs.roundUnit = minute
7. 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.
8. 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 laS propiedades que hace que se muestren los datos en formato Texto.
   1. a1.sinks.k1.hdfs.writeFormat = Text a1.sinks.k1.hdfs.fileType = DataStream