Loading Files into Hdfs Using Flume’s Spool Directory

Let’s look another way to use this flume for fetching data from local file system to HDFS.

**There are three sources for the above scenario:**

⦁ ***Exec :-*** Exec source runs a given Unix command on start-up one time and expects that process to continuously produce data on standard output location with no regular interval. If the process exits for any reason, the source also exits and will produce no further data.  
Example: PUT

⦁ Spool directory :- This source lets you insert data by placing files into a “spooling” directory on disk. This source will watch the specified directory for new files, and will parse events out of new files as they appear. After a given file has been fully read into the channel, it is renamed to indicate completion (or optionally deleted).

Unlike the Exec source, this source is reliable and will not miss data, even if Flume is restarted or killed. In exchange for this reliability, uniquely-named files must be dropped into the spooling directory

⦁ ***Netcat :-*** This source listens on a given port and turns each line of text into an Flume event and sent it via the connected channel.

**Let’s see advantages about Spool directory Flume**

⦁ It will fetch all file in local file system acting like a SOURCE and aggregating them as one single file inside HDFS.

⦁ It helps HDFS to work more efficiently, as it is good with operations on large files rather than more numbers of files.

⦁ Because of the aggregated file as mentioned in first point namenode need to keep less numbers of information as numbers of file will be less.

**NOTE:-** Hadoop Daemons should be running and can be confIrmed by “ jps ” command  
(refer the image below )



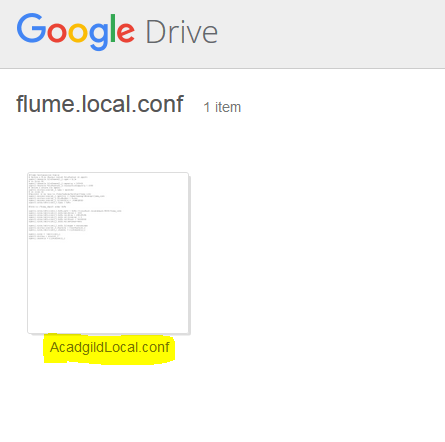
Also Flume should be downloaded and updated in .bashrc file.

**Step-by-step Demonstration : Data Streaming from Local File System to HDFS**

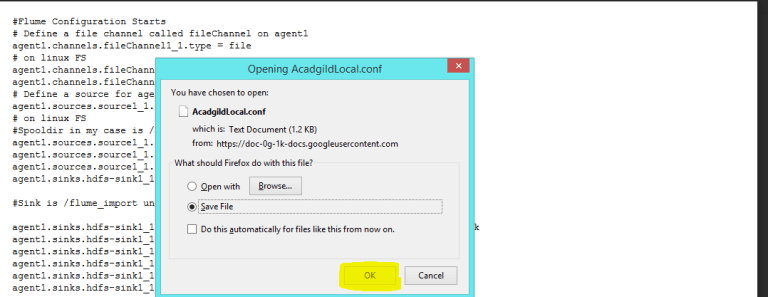
All the steps are followed by images for better reference.  
In this case we will be using spool directory as our source and HDFS as destination.  
Here we will need another agent for flume as source of data here is fetched from Local File System instead from twitter.

**Go to the below link and download the configuration file present which contains agent details.**

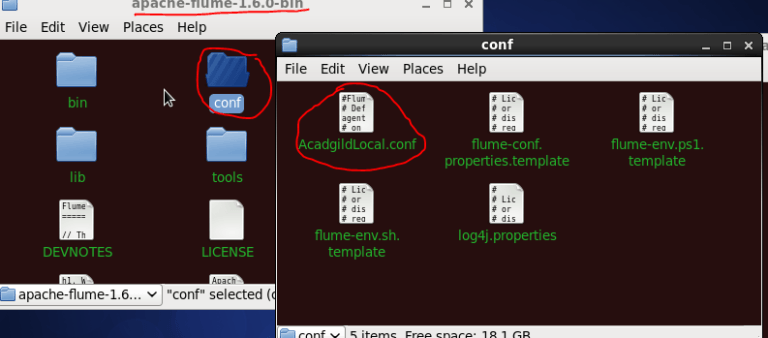
https://drive.google.com/folderview?id=0B1QaXx7tpw3SU3Eyc0pDbmx6SDg&usp=sharing



**Save the file and keep in your downloads directory.**

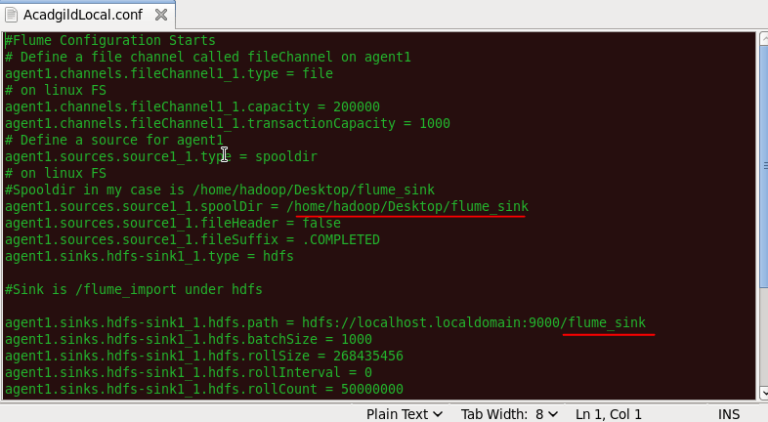


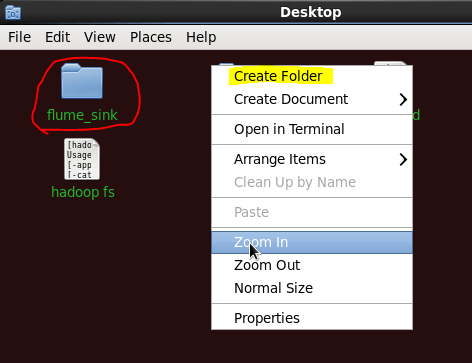
**We need to move the AcadgildLocal.conf file inside flume/conf directory.**



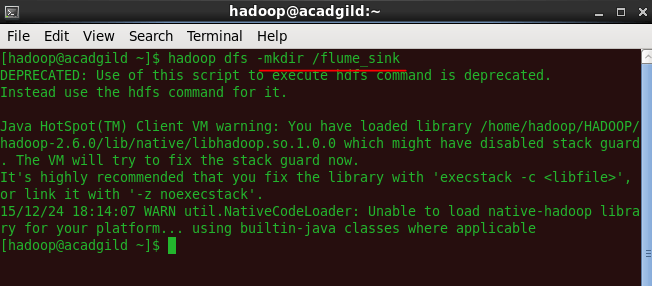
**We need to make two changes inside AcadgildLocal.conf as followed.**

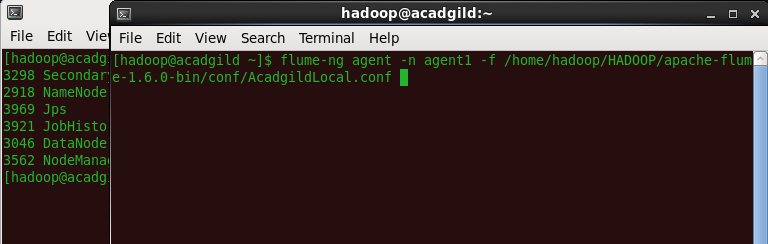
1)agent1.sources.source1\_1.spoolDir is set with input path as in local file system path.  
2)agent1.sinks.hdfs-sink1\_1.hdfs.path is set with output path as in HDFS path.

  
**Creating the folder as specified in AcadgildLocal.conf file will make our ”spooling “directory.**

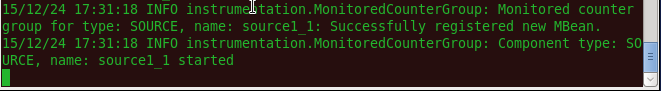


**Also we need to make destination directory inside HDFS as mentioned in AcadgildLocal.conf.**

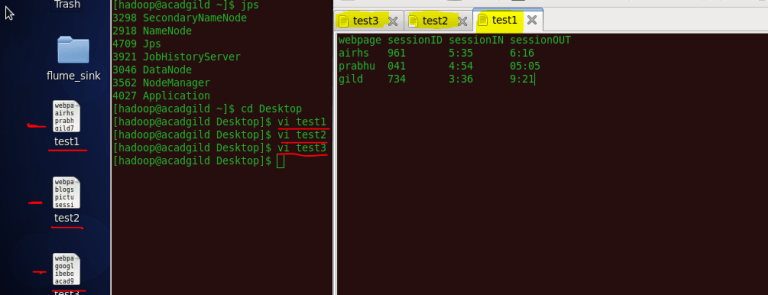
****  
**We can now open another terminal and start flume agent by the following command**  
**Command: flume-ng agent –n agent1 –f /home/hadoop/HADOOP/apache-flume-1.6.0-bin/conf/AcadgildLocal.conf**



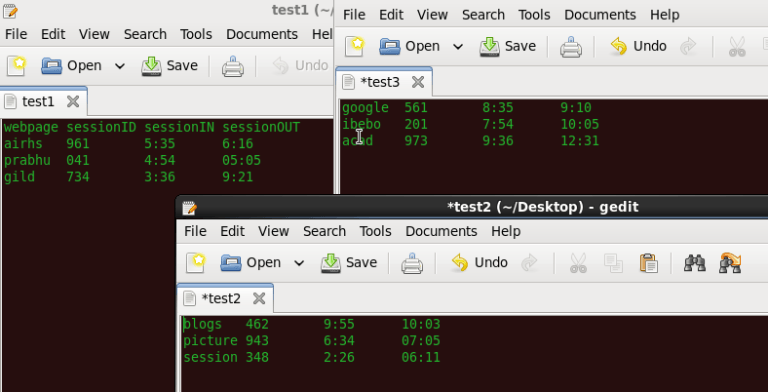
**This will conform the agent is running and we can leave this terminal running at background.**

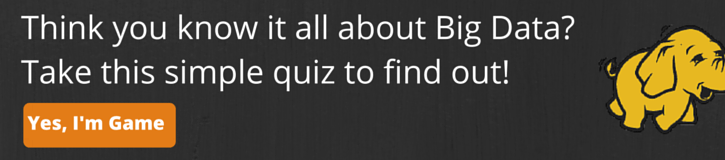


For our dummies dataset we will be creating 3 different test file which will act as 3 log files created at different times from same web server.

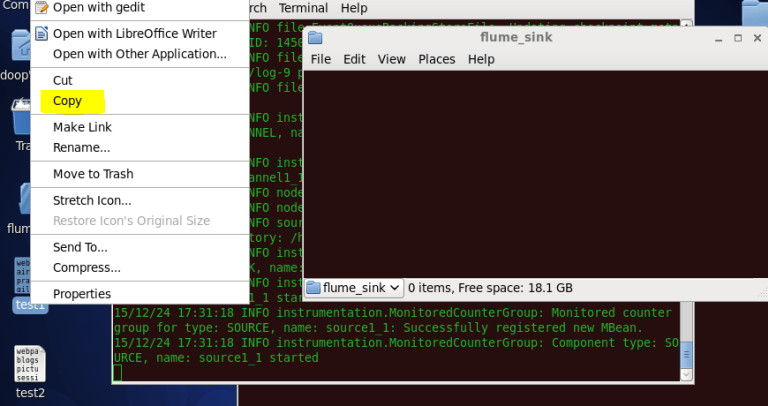


**The sample data inside the files contains list of webpage , sessionID, sessionIN, sessionOUT.**

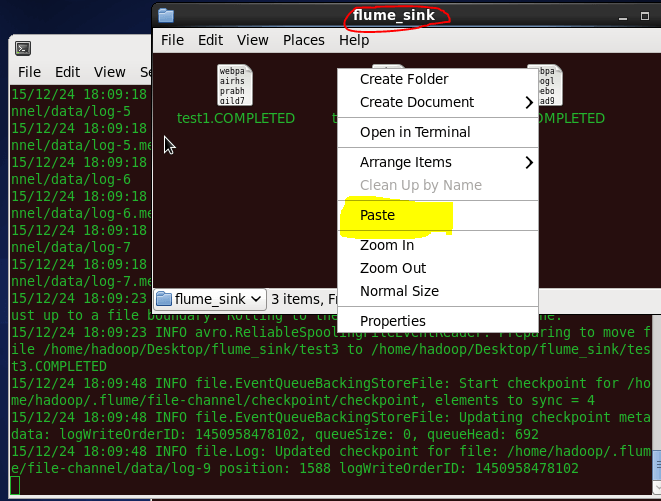


[](https://acadgild.com/blog/take-quiz-test-hadoop-knowledge-intermediate?utm_source=Blog(organic)&utm_medium=Blog%20article&utm_campaign=Big%20Data%20Landing%20Page)

Our next step needs to place our test file inside spooling directory(source).  
**We will copy the files which we created in the previous steps.**



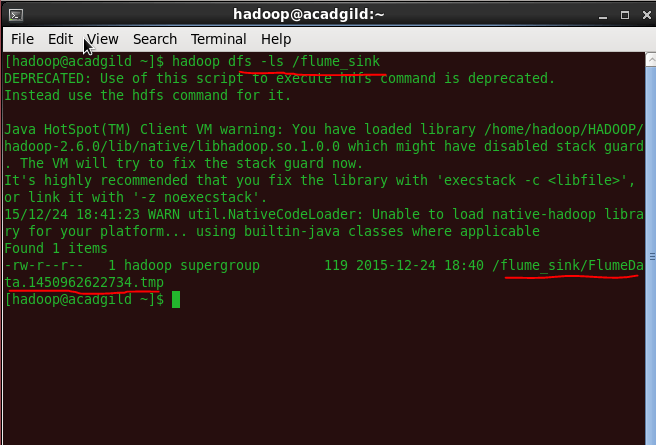
One by one we will place all the test file inside flume\_sink directory. Wait for a moment and you can see the filename change to **” filename. COMPLETED ”.**



We can now check the resultant temporary file inside HDFS at destination path generated by Flume.

By the listing command we can find only one file inside ***HDFS /flume\_sink.***

**Command:** ***hadoop dfs -ls <destination\_path>***



**We do cat to the temp file to see all the data aggregated inside one file.**

