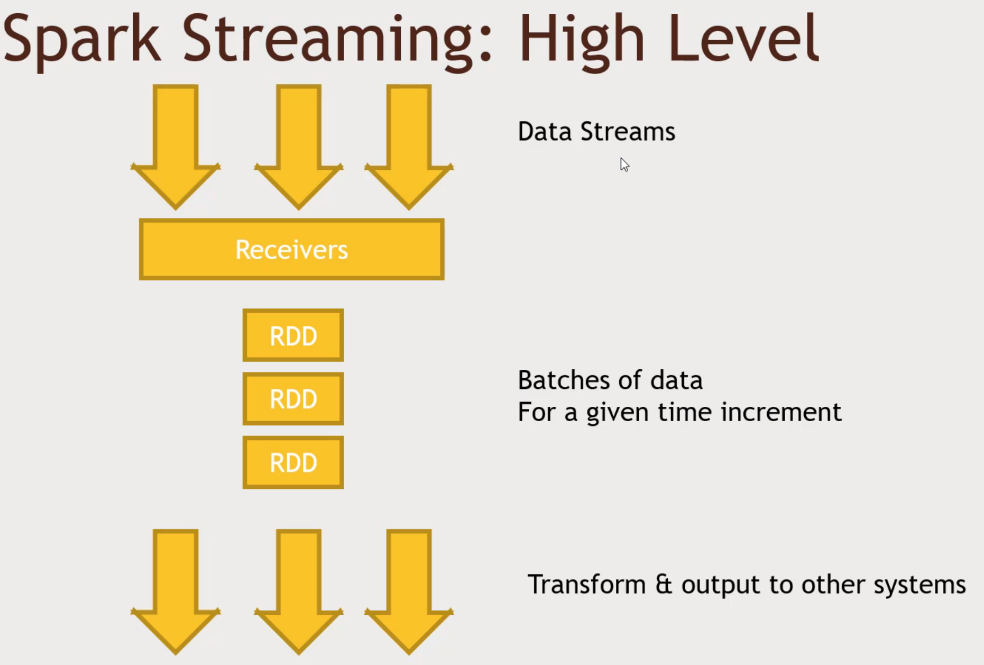
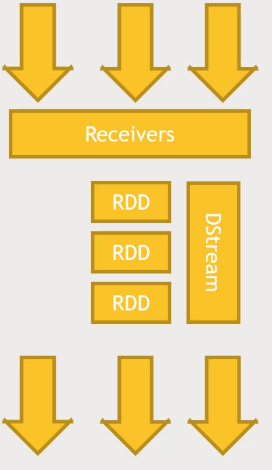
Spark Streaming: Introduction – Processing continuous inflow of data

Each RDD will contain discritized chunks. Transform & output, Micro batches, Splittin gup the data into smaller RDD. 1 Second chunks.

Work can be distributed

* Processing of RDD’s can happen in parallel on different worker nodes

DStreams (Discretized Streams)

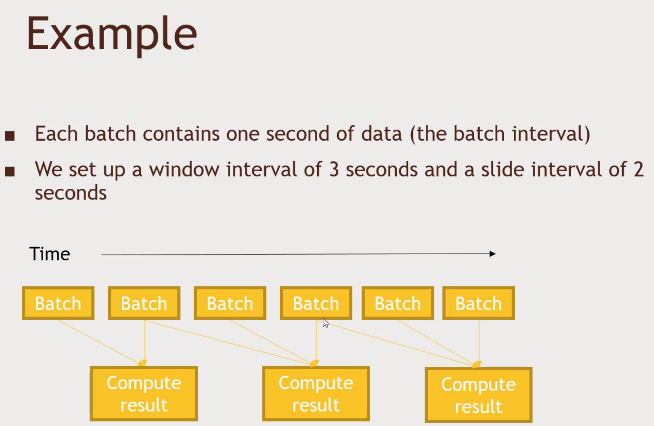
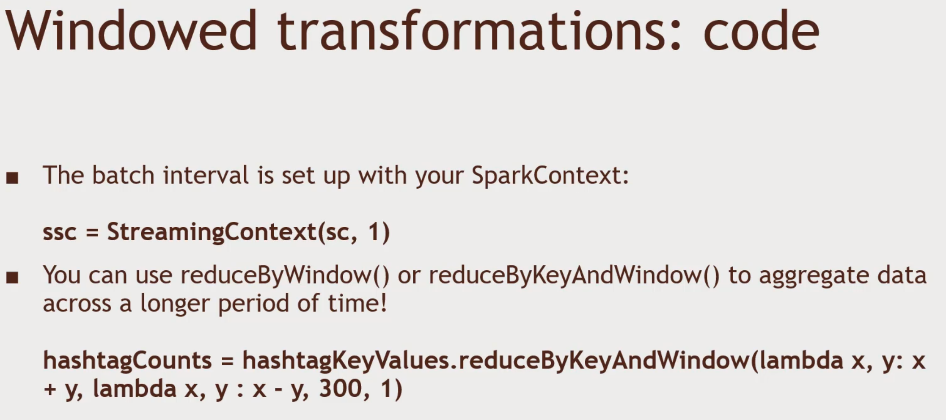
* Generates the RDD’s for each time step, and can produce output at each time step
* Can be transformed and acted on in much the same way as RDD’s
* Or you can access their underlying RDD’s if you need them.
* Map, Flatmap, Filter, reduceByKey
* Stateful data
  + You can also maintain a long-lived state on a Dstream
  + For example, running totals, broken down by keys
  + Eg: aggregating session data in web activity

Windowed Transformation

* Allow you to compute results across a longer time period than your batch interval
* Example: top-sellers from the past hour
  + You Might process data every one second(the batch interval)
  + But Maintain a window of one hour
* The window ‘slides’ as time goes on, to represent batches within the window interval

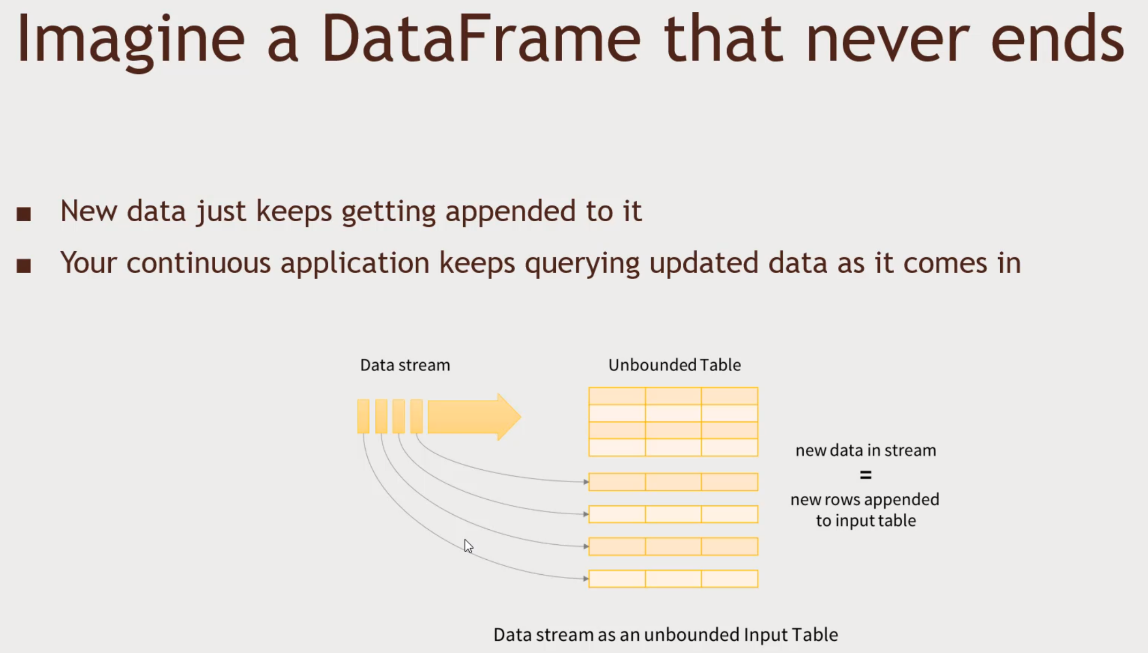
Example, analyse top seller that is received from the past hour, looking at activity data. Sliding window of one hour as time goes on. Take a snap shot of all the data. Keep the info during the window of time. Time goes on, the window will slide.

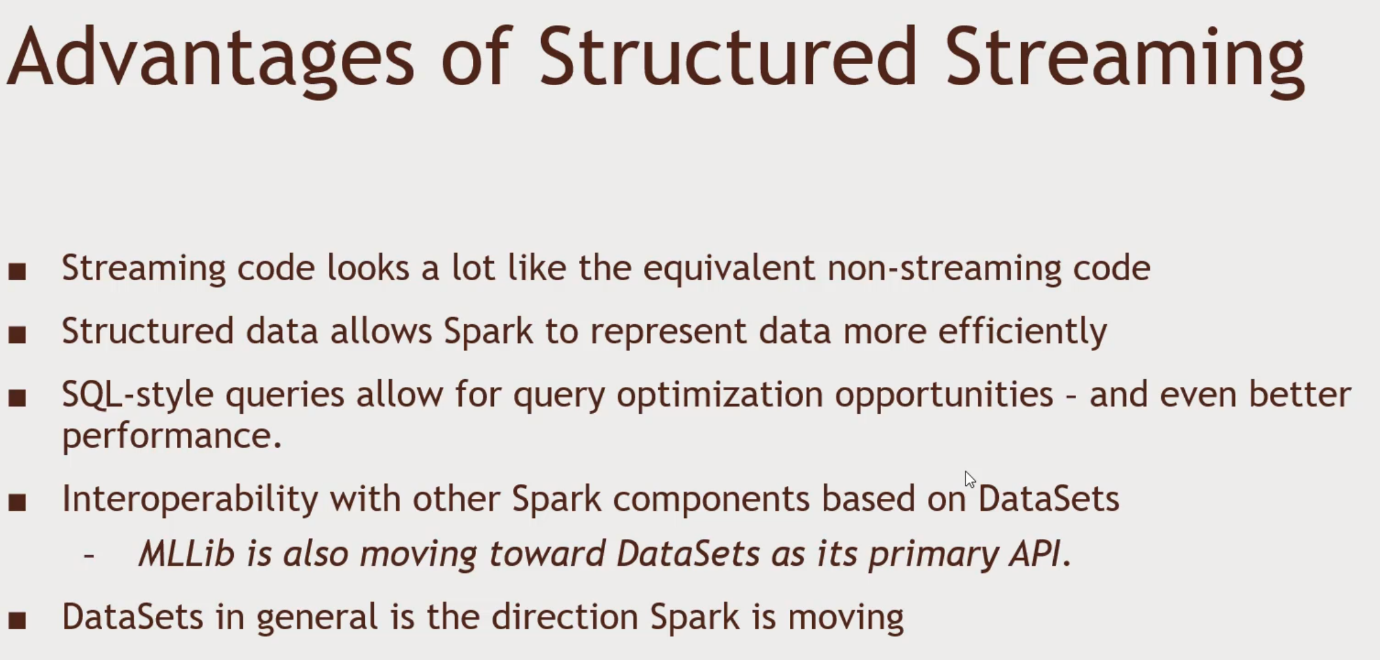
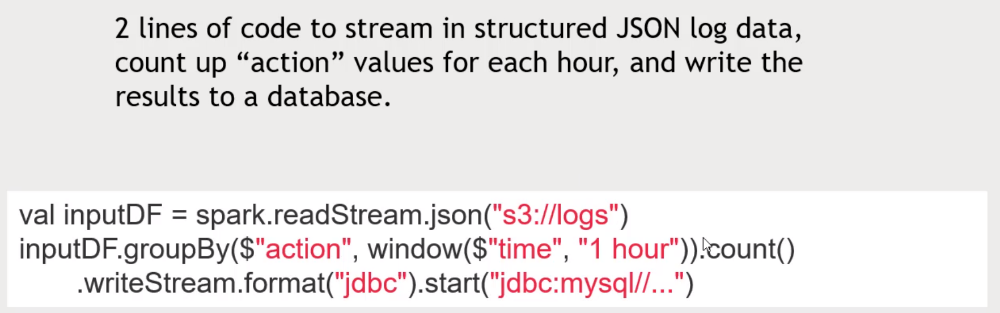
BATCH INTERVAL VS SLIDE INTERVAL VS WINDOW INTERVAL

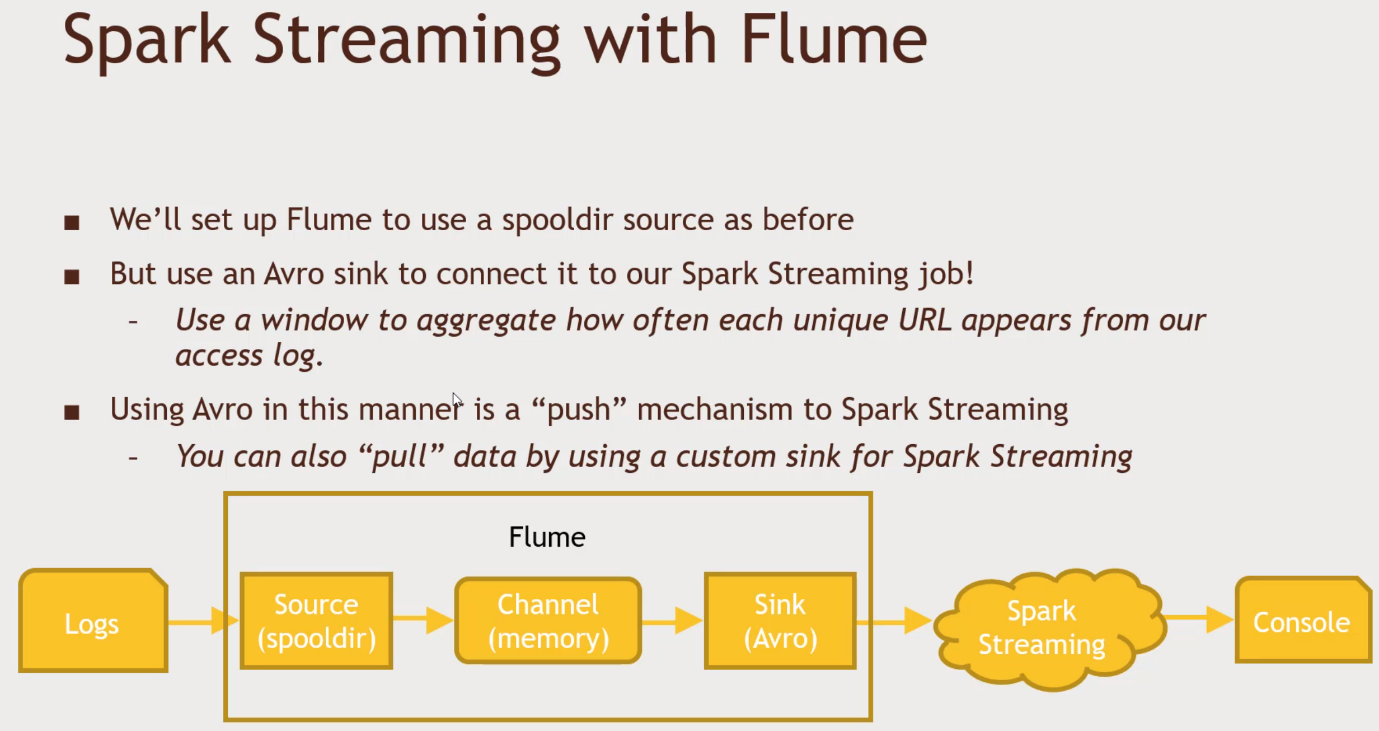
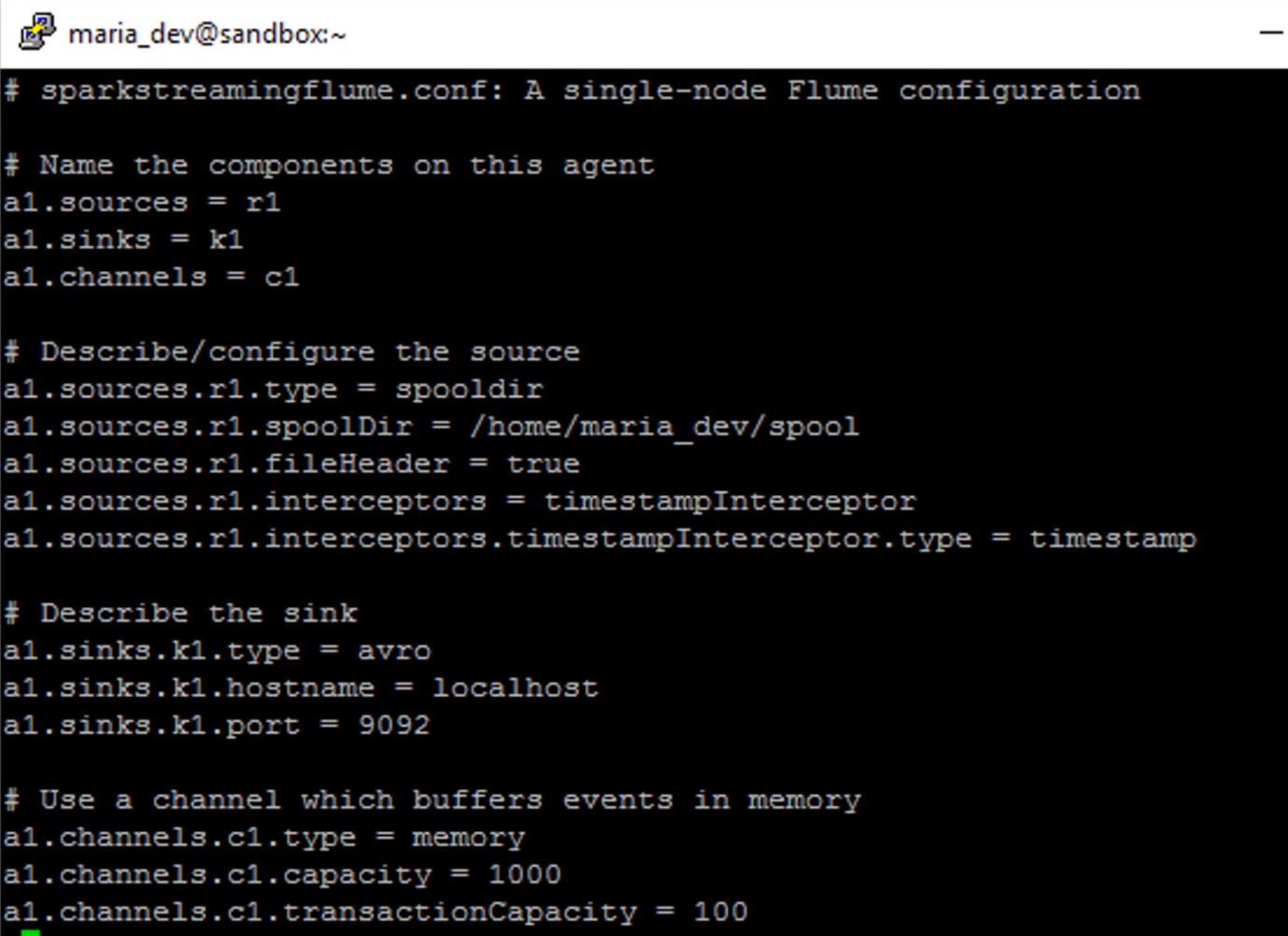
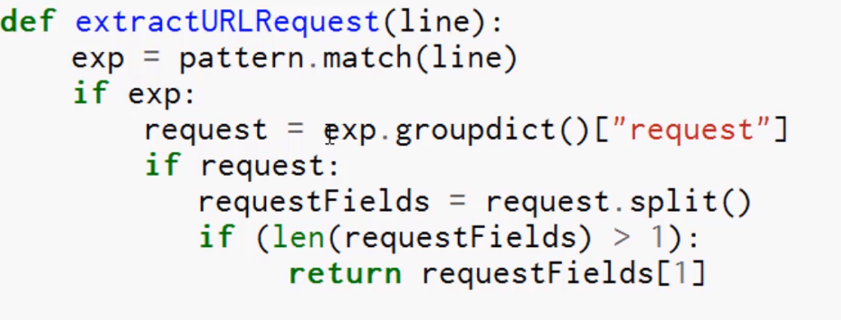
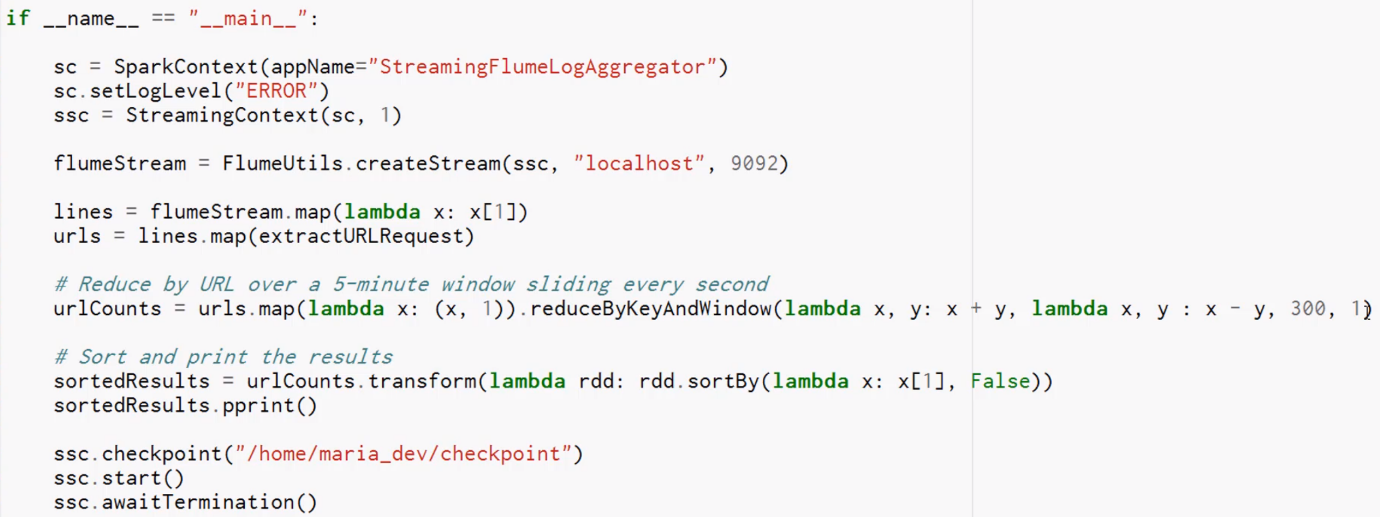
* The batch interval is how often data is captured into a Dstream
* The slide interval is how often a windowed transformation is computed
* The window interval is how far back in time the windowed transformation goes.

Given function of 2 variables, you add them up by x+y, remove by x-y, 300s window interval(5 mins), and sliding interval of 1second

Structured Streaming

* A new, higher-level API for streaming structured data
  + Available in Spark 2.0 and 2.1 as an experimental release
* Uses DataSets
  + Like a DataFrame, but with more explicit type information
  + A DataFrame is really a DataSet[Row]



1. Login to Putty and HortonWorks SandBox
2. Set up Flume to monitor spool dir and sink into Avro communication protocol
   1. wget media.sundog-soft.com/Hadoop/sparkstreamingflume.conf
   2. wget http://media.sundog-soft.com/hadoop/SparkFlume.py
   3. cat SparkFlume.py Regex Language to pattern match against each line
   4. Map function. Take a line, and match, then group that into dict and extract request field, 3 field, second one, split that out into components, and url will get plugged out of the second field
   5. The main script. Push model from Flume into spark streaming using Avro. Can set up a pull model that is custom and tailored to bidirectional data flow between flume and spark streaming

Map those url into tuple, add those all up by reduceByKeyAndWindow. Providing function to how I add and substract the things you receive, every 1 second, we will compute the url count that was seen over the past 5 minutes for each unique URL.

* 1. We need a checkpoint directory for your script whenever you run a window operation so that it can be picked up where it left off if we were to suddenly get an error.

1. Checkpoint directory
   1. mkdir checkpoint
   2. version 2. export SPARK\_MAJOR\_VERSION =2
   3. spark-submit --packages org.apache.spark:spark-streaming-flume\_2.11:2.0.0 SparkFlume.py
2. Kick off Flume (To listen to our spool directory and output data into Avro.)
   1. New putty window
   2. cd /usr/hdp/current/flume-server/
   3. bin/flume-ng agent --conf conf --conf-file ~/sparkstreamingflume.conf --name a1
3. Kick off 3rd Window
   1. ls
   2. wget <http://media.sundog-soft.com/hadoop/access_log.txt>
   3. cp access\_log.txt spool/log22.txt
   4. drop the same file into the folder.
   5. and all those numbers should double.