APACHE FLUME

A service for streaming logs into Hadoop.

Apache Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of streaming data into the Hadoop Distributed File System (HDFS). It has a simple and flexible architecture based on streaming data flows; and is robust and fault tolerant with tuneable reliability mechanisms for failover and recovery.

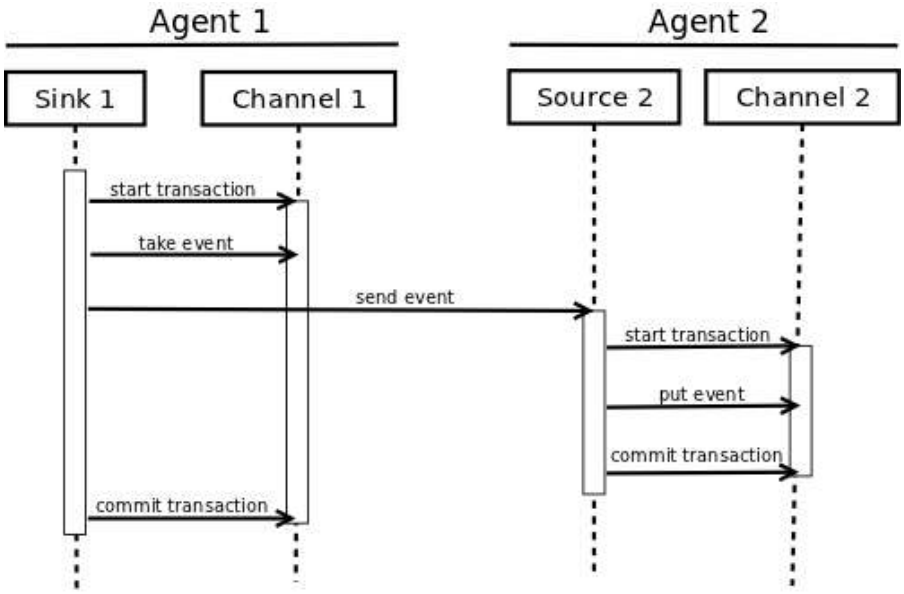
YARN coordinates data ingest from Apache Flume and other services that deliver raw data into an Enterprise Hadoop cluster.

WHAT FLUME DOES

Flume lets Hadoop users ingest high-volume streaming data into HDFS for storage. Specifically, Flume allows users to:

Feature	Description
Stream data	Ingest streaming data from multiple sources into Hadoop for storage and analysis
Insulate systems	Buffer storage platform from transient spikes, when the rate of incoming data exceeds the rate at which data can be written to the destination
Guarantee data delivery	Flume NG uses channel-based transactions to guarantee reliable message delivery. When a message moves from one agent to another, two transactions are started, one on the agent that delivers the event and the other on the agent that receives the event. This ensures guaranteed delivery semantics
Scale horizontally	To ingest new data streams and additional volume as needed

Enterprises use Flume's powerful streaming capabilities to land data from high-throughput streams in the Hadoop Distributed File System (HDFS). Typical sources of these streams are application logs, sensor and machine data, geo-location data and social media. These different types of data can be landed in Hadoop for future analysis using interactive queries in Apache Hive. Or they can feed business dashboards served ongoing data by Apache HBase.



In one specific example, Flume is used to log manufacturing operations. When one run of product comes off the line, it generates a log file about that run. Even if this occurs hundreds or thousands of times per day, the large volume log file data can stream through Flume into a tool for same-day analysis with Apache Storm or months or years of production runs can be stored in HDFS and analysed by a quality assurance engineer using Apache Hive.

HOW FLUME WORKS

Flume's high-level architecture is built on a streamlined codebase that is easy to use and extend. The project is highly reliable, without the risk of data loss. Flume also supports dynamic reconfiguration without the need for a restart, which reduces downtime for its agents.

The following components make up Apache Flume:

Component	Definition
Event	A singular unit of data that is transported by Flume (typically a single log entry)
Source	The entity through which data enters into Flume. Sources either actively poll for data or passively wait for data to be delivered to them. A variety of sources allow data to be collected, such as log4j logs and syslogs.
Sink	The entity that delivers the data to the destination. A variety of sinks allow data to be streamed to a range of destinations. One example is the HDFS sink that writes events to HDFS.
Channel	The conduit between the Source and the Sink. Sources ingest events into the channel and the sinks drain the channel.
Agent	Any physical Java virtual machine running Flume. It is a collection of sources, sinks and channels.
Client	The entity that produces and transmits the Event to the Source operating within the Agent.

Flume components interact in the following way:

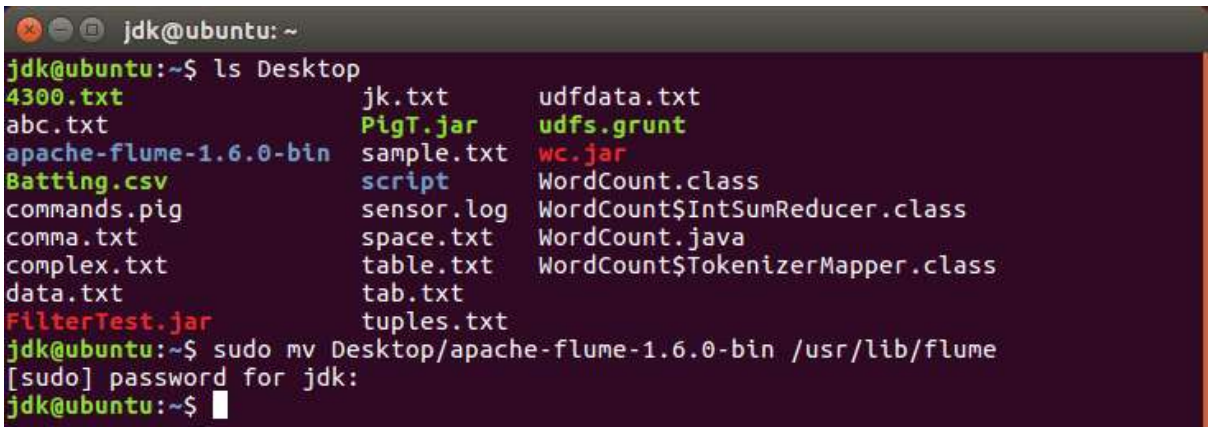
- A flow in Flume starts from the Client.
- The Client transmits the Event to a Source operating within the Agent.
- The Source receiving this Event then delivers it to one or more Channels.
- One or more Sinks operating within the same Agent drains these Channels.
- Channels decouple the ingestion rate from drain rate using the familiar producer-consumer model of data exchange.
- When spikes in client side activity cause data to be generated faster than can be handled by the provisioned destination capacity can handle, the Channel size increases. This allows sources to continue normal operation for the duration of the spike.
- The Sink of one Agent can be chained to the Source of another Agent. This chaining enables the creation of complex data flow topologies.

Because Flume's distributed architecture requires no central coordination point. Each agent runs independently of others with no inherent single point of failure, and Flume can easily scale horizontally.

TWITTER DATA FETCHING AND PROCESSING

Installing Flume

1. First we will install flume in the system. Copy the tar file of the apache flume and then extract it on the desktop. Now we move this extracted file from the desktop to the destination /usr/lib/flume using the sudo mv command.



2. To check if the file has been moved, we use the command ls. Now open the bashrc file as shown.

```
jdk@ubuntu:~$ ls Desktop
4300.txt        data.txt         sensor.log    udfs.grunt
abc.txt         FilterTest.jar   space.txt     wc.jar
Batting.csv     jk.txt           table.txt     WordCount.class
commands.pig    PigT.jar         tab.txt       WordCount$IntSumReducer.class
comma.txt       sample.txt       tuples.txt    WordCount.java
complex.txt     script           udfdata.txt   WordCount$TokenizerMapper.class
jdk@ubuntu:~$ gedit ~/.bashrc
jdk@ubuntu:~$
```

3. We add the following lines in this file for configuration purposes:

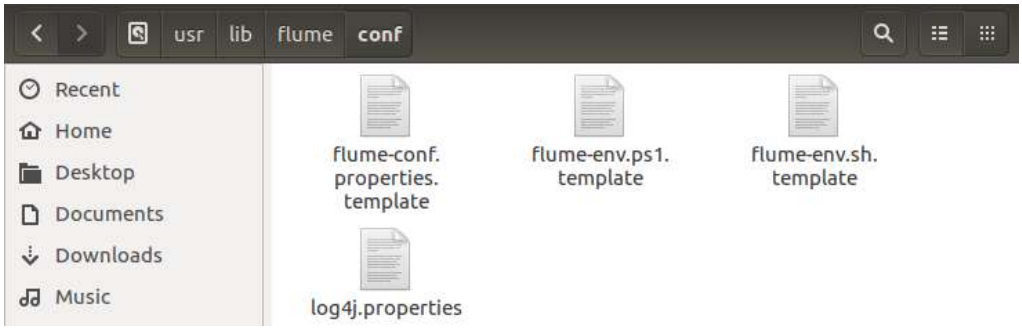
```
export FLUME_HOME=/usr/lib/flume
export FLUME_CONF_DIR=$FLUME_HOME/conf
export FLUME_CLASSPATH=$FLUME_CONF_DIR
export PATH=$PATH:$FLUME_HOME/bin
```

Open ▾ Save

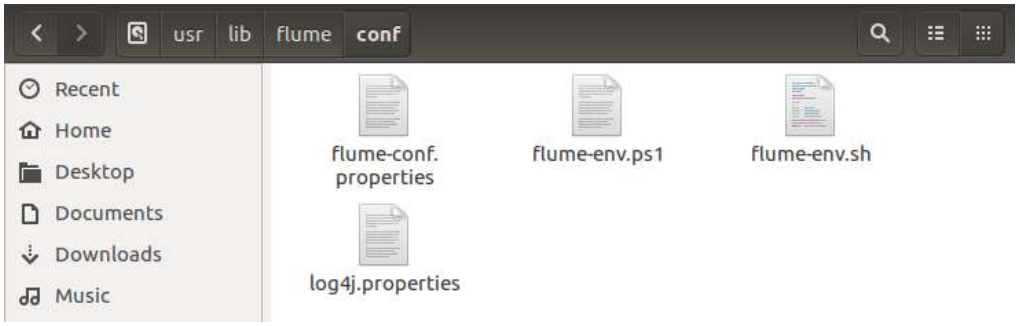
```
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_INSTALL=/usr/local/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_CLASSPATH=$JAVA_HOME/lib/tools.jar
export SQOOP_HOME=/usr/local/pig
export PATH=$PATH:$SQOOP_HOME/bin
export HIVE_HOME=/usr/local/hive
export PATH=$PATH:$HIVE_HOME/bin
export SQOOP_HOME=/usr/lib/sqoop
export PATH=$PATH:$SQOOP_HOME/bin
export FLUME_HOME=/usr/lib/flume
export FLUME_CONF_DIR=$FLUME_HOME/conf
export FLUME_CLASSPATH=$FLUME_CONF_DIR
export PATH=$PATH:$FLUME_HOME/bin
```

sh ▾ Tab Width: 8 ▾ Ln 137, Col 34 ▾ INS

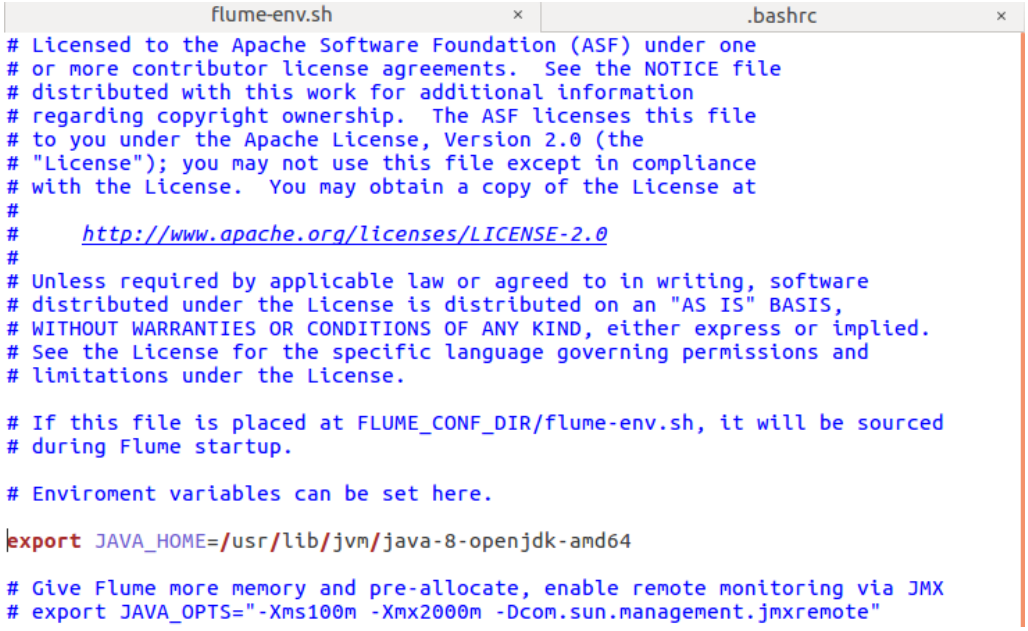
4. Now, we change the names of 3 files (/usr/lib/flume/conf/) flume-conf.properties.template, flume-env.ps1.template and flume-env.sh.template renaming them and removing the string 'template' from their name.



5. This is the output of the same.



6. Now, we open the flume-env.sh file and add the JAVA_HOME PATH in it.

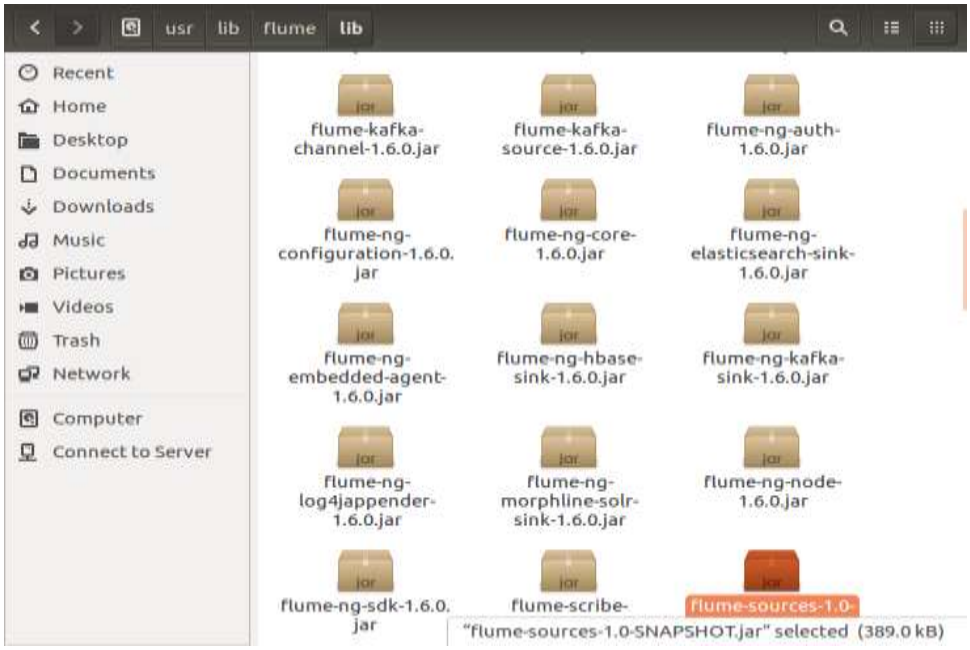


7. Now we permanently save the bashrc file.

8. We use the flume-ng command. If the help desk opens, then flume has been successfully installed.

```
jdk@ubuntu: ~
flume-version: command not found
jdk@ubuntu:~$ flume-ng
Error: Unknown or unspecified command ''

Usage: /usr/lib/flume/bin/flume-ng <command> [options]...

commands:
  help                      display this help text
  agent                     run a Flume agent
  avro-client               run an avro Flume client
  version                   show Flume version info

global options:
  --conf,-c <conf>          use configs in <conf> directory
  --classpath,-C <cp>       append to the classpath
  --dryrun,-d               do not actually start Flume, just print the command
  --plugins-path <dirs>     colon-separated list of plugins.d directories. See t
he
                            plugins.d section in the user guide for more details
.
                            Default: $FLUME_HOME/plugins.d
  -Dproperty=value          sets a Java system property value
  -Xproperty=value          sets a Java -X option
```

9. Now we move to /usr/lib/flume/lib and copy a jar file named flume-sources-1.0-SNAPSHOT.jar (External Files folder) into this directory.



10. After that we open the flume-env.ps1 (/usr/lib/flume/conf) file and add the FLUME_CLASSPATH in it.

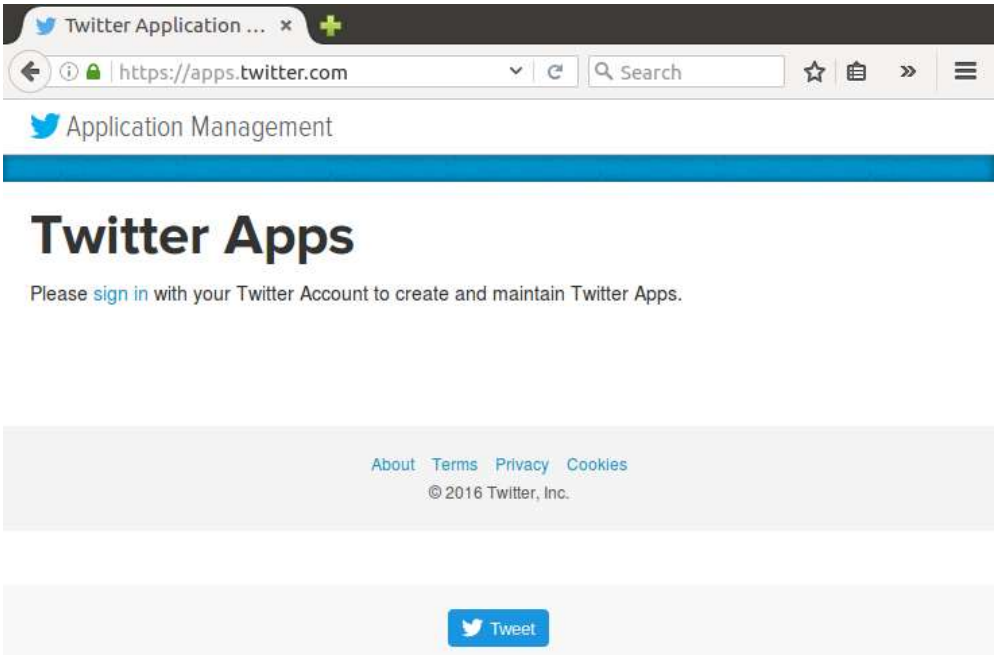
```
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements.  See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership.  The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.


# Give Flume more memory and pre-allocate, enable remote monitoring via JMX
$JAVA_OPTS="-Xms100m -Xmx200m -Dcom.sun.management.jmxremote"

# Foll. classpath will be included in Flume's classpath.
# Note that the Flume conf directory is always included in the classpath.
   # Example:  "path1;path2;path3"
$FLUME_CLASSPATH="usr/lib/flume/lib/flume-sources-1.0-SNAPSHOT.jar"
```

Open Save

Saving file '/usr/lib/flume/conf/flu... Plain Text Tab Width: 8 Ln 24, Col 67 INS

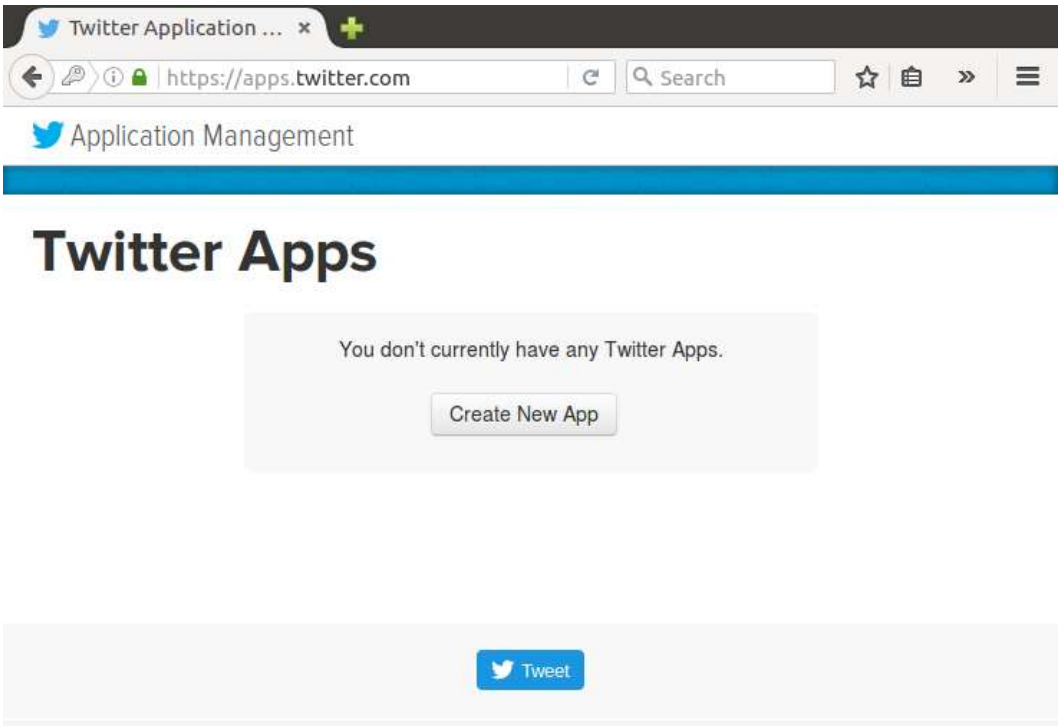
Creating Twitter Developer Application

11. Now we open the website dev.twitter.com/apps in the Mozilla Firefox Browser.

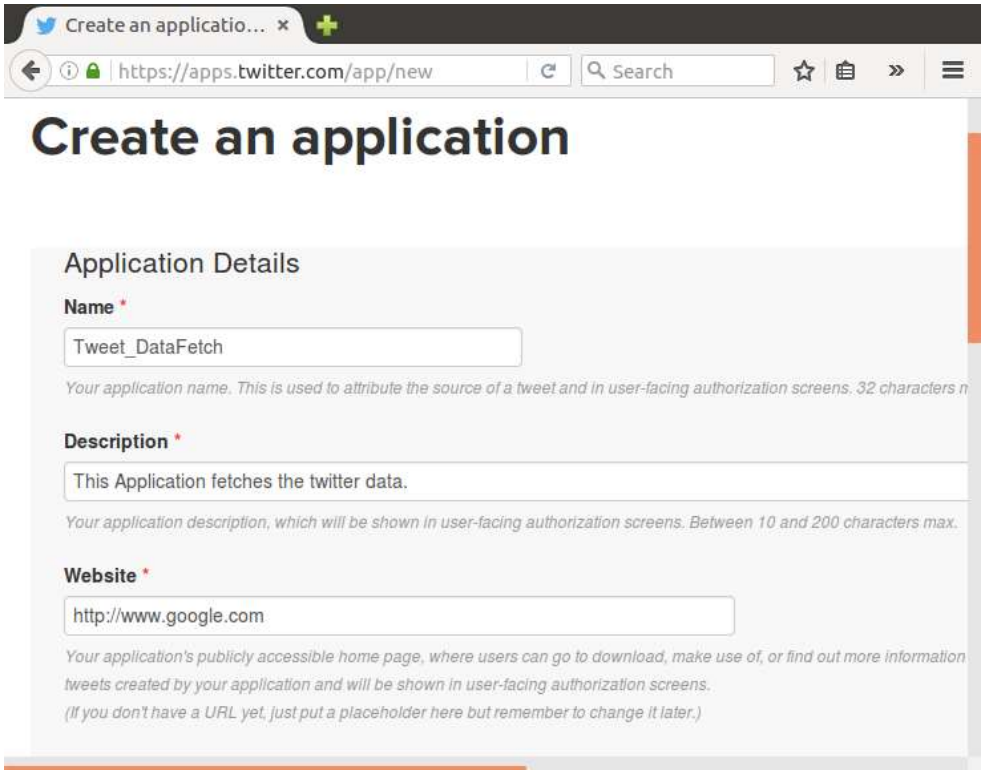


12. We will now see the website suggesting us to sign in. So, we sign into our twitter account.

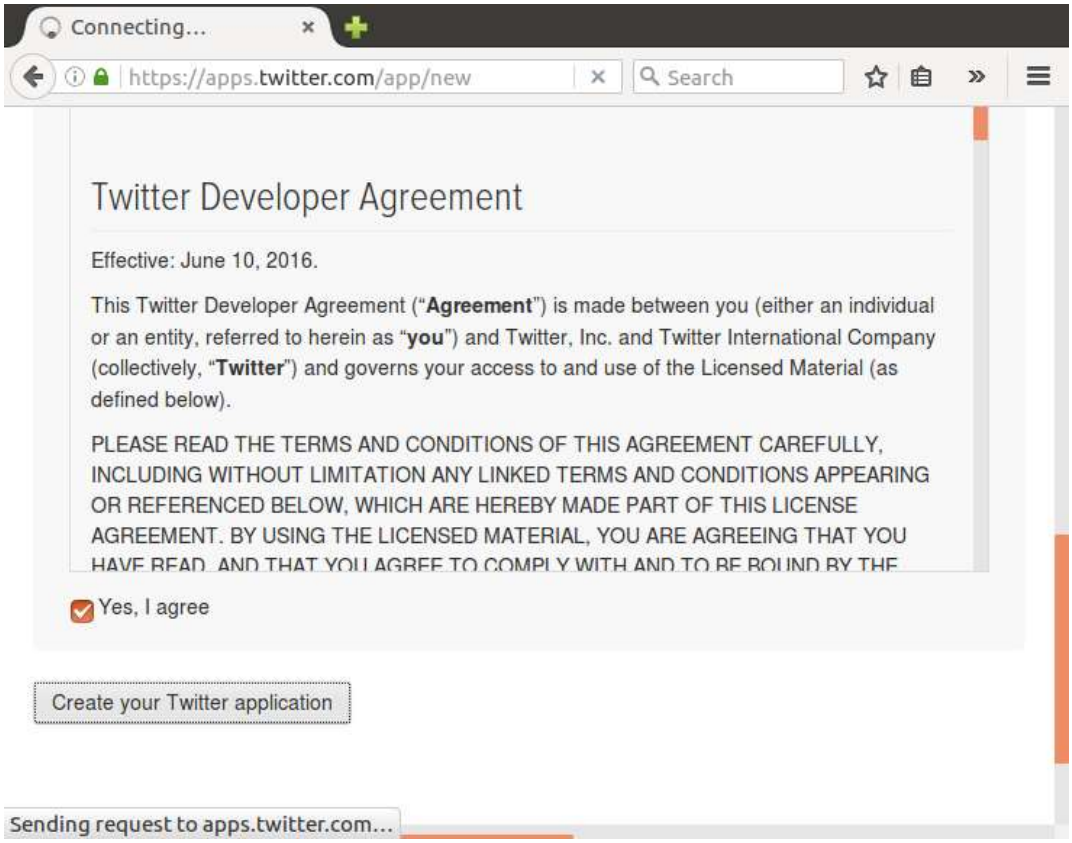
    Click on Create New App.



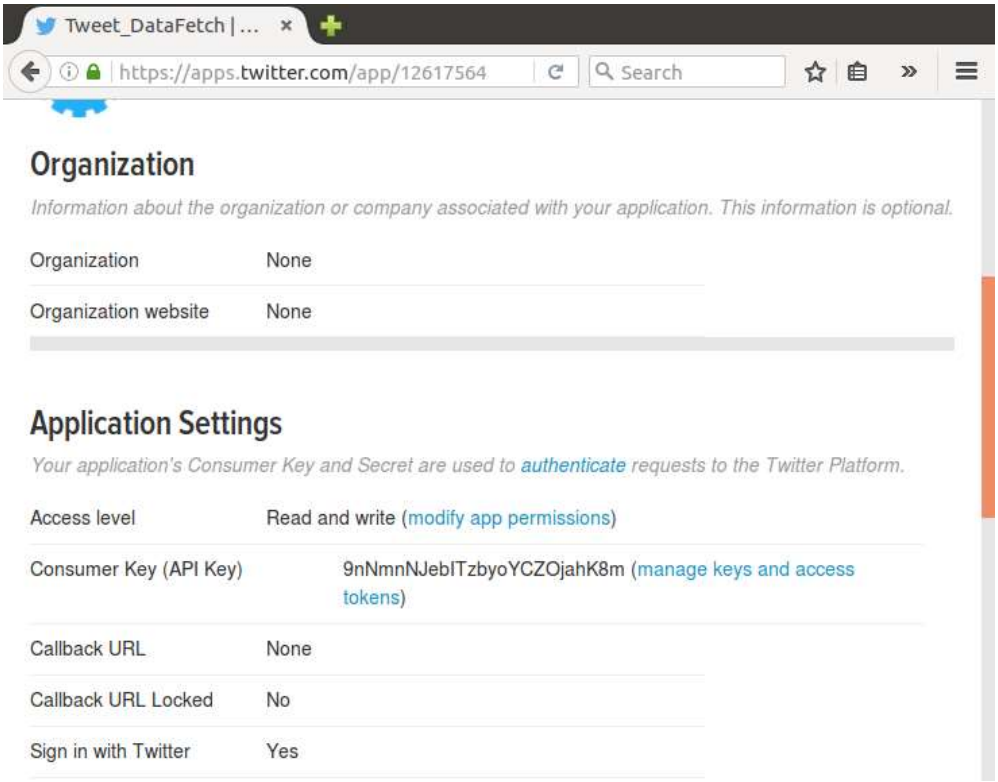
13. Fill in all the required fields to make the application and use the website as google.com.



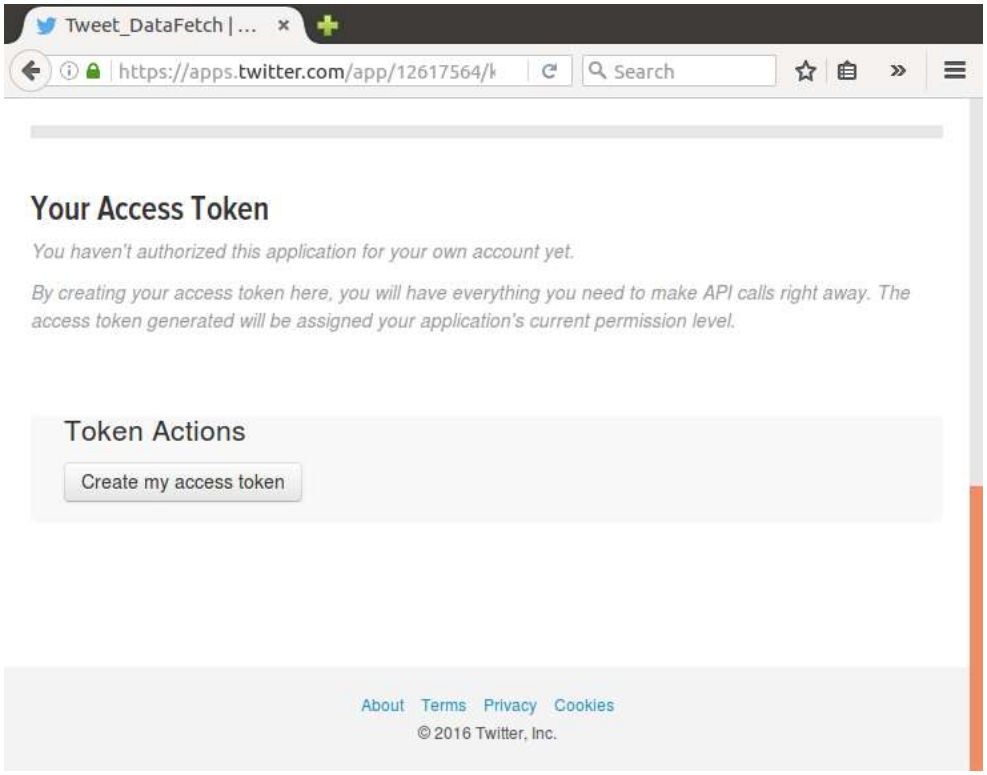
14. Now, scroll down and tick the option Yes, I agree and then click Create your Twitter application.



15. Click on manage keys and access tokens.



16. Now click on Create my access token.



17. Now, we copy the file flume.conf (External Files folder) to /usr/lib/flume/conf/ and open it. Then change the following keys in the file. These keys will be obtained from the Step 16.

18. These are the keys which we will change in the flume.conf file:
    - Access Token
    - Access Token Secret
    - Consumer Key (API Key)
    - Consumer Secret (API Secret)

Note that the below mentions fields are unique for every single twitter application.

19. The keys are changed in this. Also add the Keywords that are needed to be extracted from twitter.

Note that he Highlighted Fields are required to be modified.

Open ▾ Save

```
#sandbox.sinks.sink_to_hdfs.channel = file_channel

TwitterAgent.sources = Twitter
TwitterAgent.channels = MemChannel
TwitterAgent.sinks = HDFS

TwitterAgent.sources.Twitter.type = com.cloudera.flume.source.TwitterSource
TwitterAgent.sources.Twitter.channels = MemChannel
TwitterAgent.sources.Twitter.consumerKey = 9nNmnNJebITzbyoYCZOjahK8m
TwitterAgent.sources.Twitter.consumerSecret =
XmSIlUZqeG8Z4a2cx41GUXZdzqociGzb4J619JEclMMrIFsGkI
TwitterAgent.sources.Twitter.accessToken = 754950755547619329-
PLoMGCDHXuT2ykdPVvCDUP6xkc44Ni6
TwitterAgent.sources.Twitter.accessTokenSecret =
1p5vC7XGJnRa7xrP4YiIO5RG3xAmYYoEh1iR4t4D469zu

TwitterAgent.sources.Twitter.keywords = virat,kohli

TwitterAgent.sinks.HDFS.channel = MemChannel
TwitterAgent.sinks.HDFS.type = hdfs
TwitterAgent.sinks.HDFS.hdfs.path =/flumedir/data/tweets_raw
TwitterAgent.sinks.HDFS.hdfs.fileType = DataStream
TwitterAgent.sinks.HDFS.hdfs.writeFormat = Text
TwitterAgent.sinks.HDFS.hdfs.batchSize = 1000
TwitterAgent.sinks.HDFS.hdfs.rollSize = 0
TwitterAgent.sinks.HDFS.hdfs.rollCount = 10000

TwitterAgent.channels.MemChannel.type = memory
TwitterAgent.channels.MemChannel.capacity = 10000
TwitterAgent.channels.MemChannel.transactionCapacity = 100
```

Plain Text ▾ Tab Width: 8 ▾ Ln 44, Col 61 ▾ INS

Extracting Data from Twitter

20. Start all the services using the start-all.sh command.

```
jdk@ubuntu:~$ jps
3979 Jps
jdk@ubuntu:~$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-jdk-namen
ode-ubuntu.out
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-jdk-datan
ode-ubuntu.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-jd
k-secondarynamenode-ubuntu.out
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-jdk-resourceman
ager-ubuntu.out
localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-jdk-node
manager-ubuntu.out
jdk@ubuntu:~$
```

21. Now, make the following new directories:

```
jdk@ubuntu:~$ hadoop fs -mkdir /flumedir/data
jdk@ubuntu:~$ hadoop fs -mkdir /flumedir/data/dictionary
jdk@ubuntu:~$ hadoop fs -mkdir /flumedir/data/time_zone_map
jdk@ubuntu:~$ hadoop fs -mkdir /flumedir/data/tweets_raw
jdk@ubuntu:~$ hadoop fs -ls /flumedir/data
Found 3 items
drwxr-xr-x   - jdk supergroup          0 2016-07-23 21:35 /flumedir/data/diction
ary
drwxr-xr-x   - jdk supergroup          0 2016-07-23 21:35 /flumedir/data/time_zo
ne_map
drwxr-xr-x   - jdk supergroup          0 2016-07-23 21:36 /flumedir/data/tweets_
raw
jdk@ubuntu:~$
```

22. Then we copy the files dictionary.tsv and time_zone_map.tsv (from External Files folder) to their respective directories in HDFS:

```
jdk@ubuntu:~$ hadoop fs -put Desktop/dictionary.tsv /flumedir/data/dictionary
jdk@ubuntu:~$ hadoop fs -put Desktop/time_zone_map.tsv /flumedir/data/time_zone_
map
jdk@ubuntu:~$ hadoop fs -ls -R /flumedir/data
drwxr-xr-x   - jdk supergroup          0 2016-07-23 21:41 /flumedir/data/diction
ary
-rw-r--r--   1 jdk supergroup     308921 2016-07-23 21:41 /flumedir/data/diction
ary/dictionary.tsv
drwxr-xr-x   - jdk supergroup          0 2016-07-23 21:42 /flumedir/data/time_zo
ne_map
-rw-r--r--   1 jdk supergroup       3021 2016-07-23 21:42 /flumedir/data/time_zo
ne_map/time_zone_map.tsv
drwxr-xr-x   - jdk supergroup          0 2016-07-23 21:36 /flumedir/data/tweets_
raw
jdk@ubuntu:~$
```

23. We will now start the flume agent using the following command:

```
/usr/lib/flume/bin/flume-ng agent --conf ./conf/ -f /usr/lib/flume/conf/flume.conf -Dflume.root.logger=DEBUG,console -n TwitterAgent
```

```
jdk@ubuntu:~$ /usr/lib/flume/bin/flume-ng agent --conf ./conf/ -f /usr/lib/flume
/conf/flume.conf -Dflume.root.logger=DEBUG,console -n TwitterAgent
Info: Including Hadoop libraries found via (/usr/local/hadoop/bin/hadoop) for HD
FS access
Info: Excluding /usr/local/hadoop/share/hadoop/common/lib/slf4j-api-1.7.10.jar f
rom classpath
Info: Excluding /usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.10.j
ar from classpath
Info: Including Hive libraries found via (/usr/local/hive) for Hive access
+ exec /usr/lib/jvm/java-8-openjdk-amd64/bin/java -Xmx20m -Dflume.root.logger=DE
BUG,console -cp './conf/:/usr/lib/flume/lib/*:/usr/local/hadoop/etc/hadoop:/usr/
local/hadoop/share/hadoop/common/lib/activation-1.1.jar:/usr/local/hadoop/share/
hadoop/common/lib/apacheds-i18n-2.0.0-M15.jar:/usr/local/hadoop/share/hadoop/com
mon/lib/apacheds-kerberos-codec-2.0.0-M15.jar:/usr/local/hadoop/share/hadoop/com
```

24. This is the list of twitter data extracted which contains the keyword as specified in the flume.conf file.

Browsing HDFS

localhost:50070/explorer.html#/flumedir/da

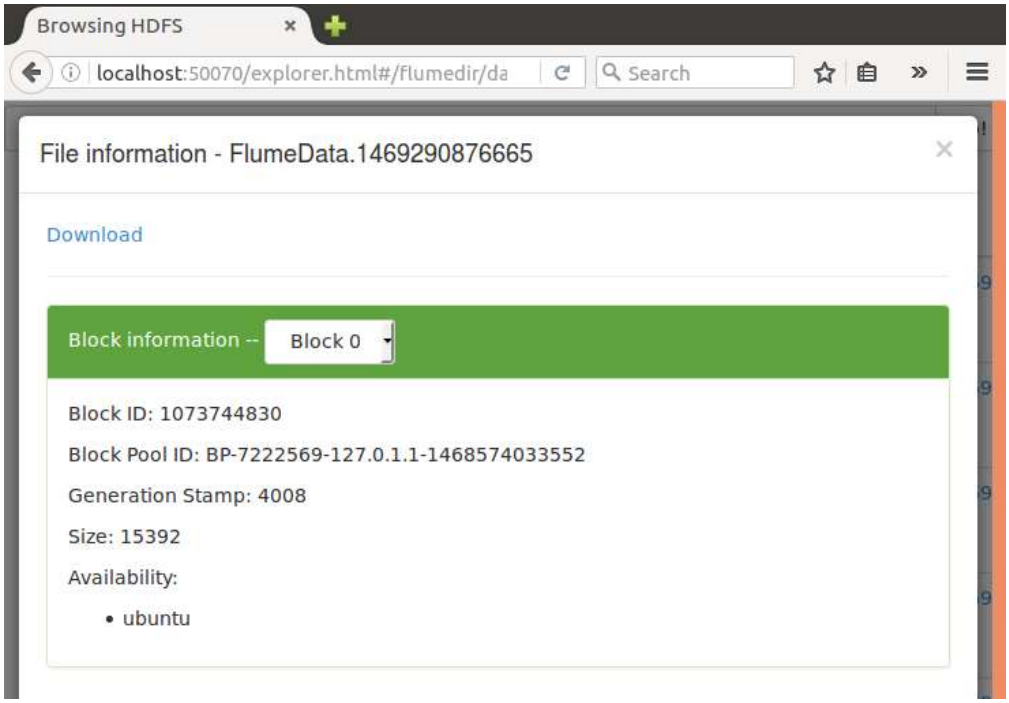
Search

/flumedir/data/tweets_raw

Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	jdk	supergroup	15.03 KB	7/23/2016, 9:51:50 PM	1	128 MB	FlumeData.1469
-rw-r--r--	jdk	supergroup	8.17 KB	7/23/2016, 9:52:32 PM	1	128 MB	FlumeData.1469
-rw-r--r--	jdk	supergroup	12.92 KB	7/23/2016, 9:53:24 PM	1	128 MB	FlumeData.1469
-rw-r--r--	jdk	supergroup	16.65 KB	7/23/2016, 9:53:56 PM	1	128 MB	FlumeData.1469
-rw-r--r--	jdk	supergroup	26.87 KB	7/23/2016, 9:54:32 PM	1	128 MB	FlumeData.1469

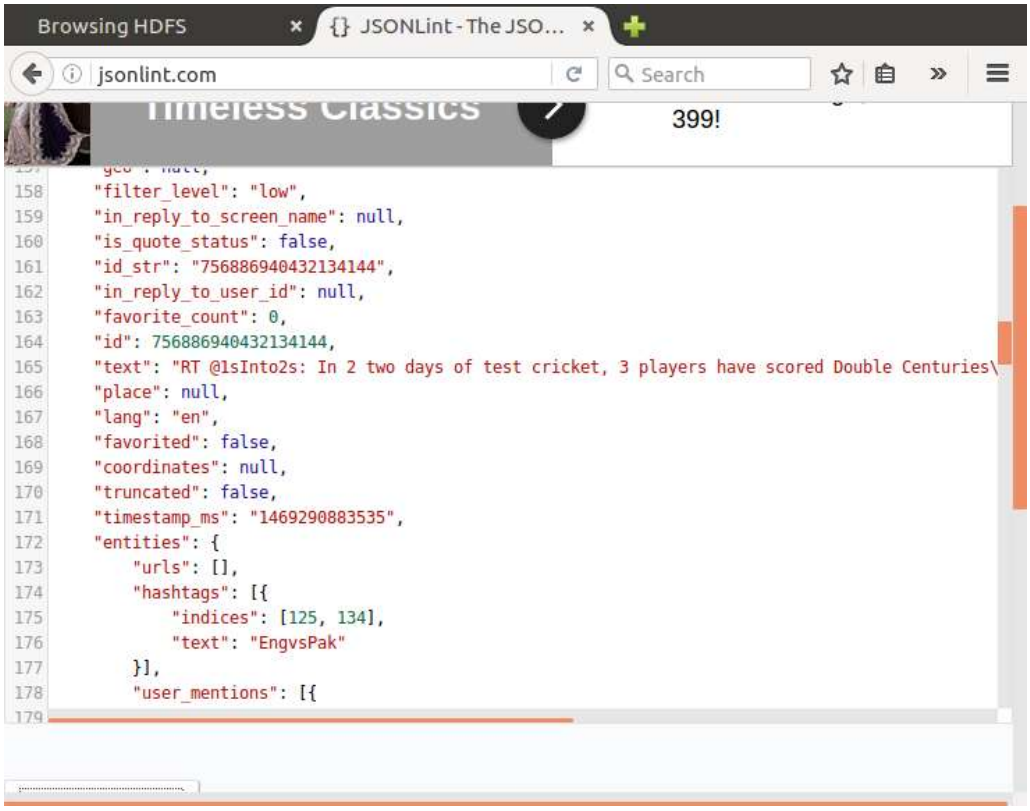
25. We can check the files by downloading them and seeing the tweets relating to the keyword.



26. The file downloaded will be like this:

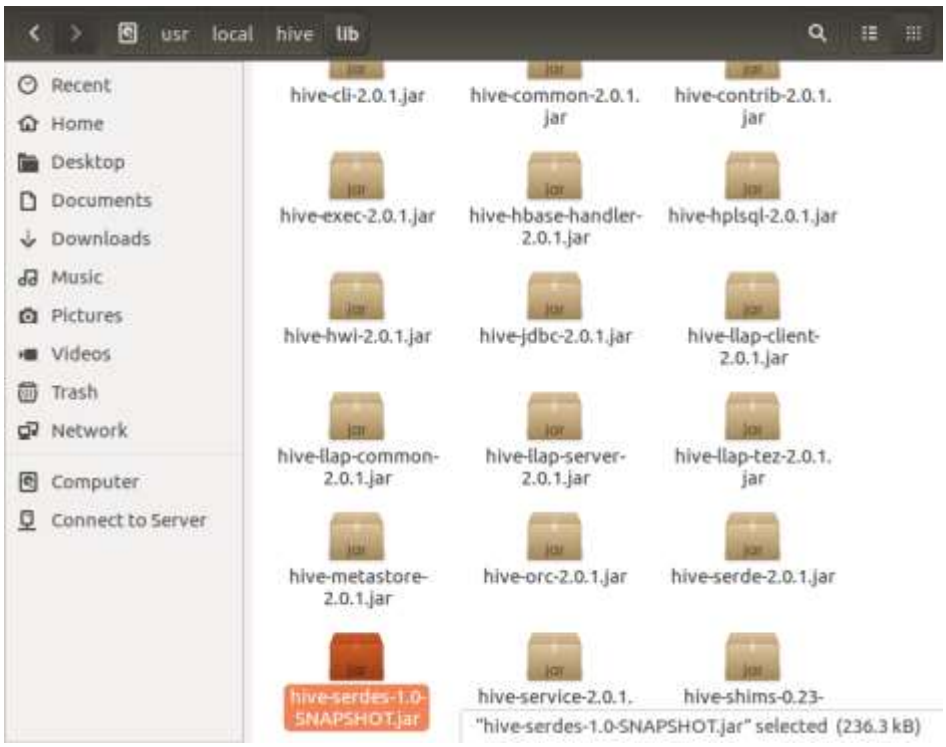
27. If we clean this downloaded file using the JSON Validator we will see the tweets in a human readable format as shown:

Then click on Validate JSON. The output that will become is as follows:



Performing Analysis using Hive

28. Now we copy the hive-serdes-1.0-SNAPSHOT.jar (from External Files folder) file into the directory /usr/local/hive/lib. This will be used by the hive shell to extract the clean data from the downloaded data into the hive table.



29. Now we create a file tweets.sql which is on the Desktop.

Note: tweets.sql is a SQL script file that will perform the analysis on the Tweet raw data.

Open ▾ Save

```sql
drop table if exists time_zone_map;
drop table if exists Mytweets_raw;
drop table if exists tweets_sentiment;
drop table if exists tweetsbl;
drop table if exists dictionary;
drop table if exists tweetsbiaggr;
drop table if exists tweetcompare;

drop view if exists tweets_clean;
drop view if exists tweets_simple;
drop view if exists l1;
drop view if exists l2;
drop view if exists l3;
drop view if exists A;
drop view if exists B;
drop view if exists C;

ADD JAR /usr/local/hive/lib/hive-serdes-1.0-SNAPSHOT.jar;

--create the tweets_raw table containing the records as received from Twitter

CREATE EXTERNAL TABLE Mytweets_raw (
   id BIGINT,
   created_at STRING,
   source STRING,
   favorited BOOLEAN,
   retweet_count INT,
   retweeted_status STRUCT<
      text:STRING,
      tuser:STRUCT<screen_name:STRING,name:STRING>>,
```

SQL ▾ Tab Width: 8 ▾ Ln 147, Col 1 ▾ INS

Open ▾ Save

```sql
   entities STRUCT<
      urls:ARRAY<STRUCT<expanded_url:STRING>>,
      user_mentions:ARRAY<STRUCT<screen_name:STRING,name:STRING>>,
      hashtags:ARRAY<STRUCT<text:STRING>>>,
   text STRING,
   tuser STRUCT<
      screen_name:STRING,
      name:STRING,
      friends_count:INT,
      followers_count:INT,
      statuses_count:INT,
      verified:BOOLEAN,
      utc_offset:INT,
      time_zone:STRING>,
   in_reply_to_screen_name STRING
)
ROW FORMAT SERDE 'com.cloudera.hive.serde.JSONSerDe'
LOCATION '/flumedir/data/tweets_raw';

load data inpath '/flumedir/data/tweets_raw/FlumeData.*' INTO TABLE Mytweets_raw;

-- create sentiment dictionary
CREATE EXTERNAL TABLE dictionary (
    type string,
    length int,
    word string,
    pos string,
    stemmed string,
    polarity string
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE
LOCATION '/flumedir/data/dictionary';

-- loading data to the table dictionary
load data inpath '/flumedir/data/dictionary/dictionary.tsv' INTO TABLE
dictionary;

CREATE EXTERNAL TABLE time_zone_map (
    time_zone string,
    country string
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE
LOCATION '/flumedir/data/time_zone_map';

-- loading data to the table time_zone_map
load data inpath '/flumedir/data/time_zone_map/time_zone_map.tsv' INTO TABLE
time_zone_map;


-- Clean up tweets
CREATE VIEW tweets_simple AS
SELECT
  id,
  cast ( from_unixtime( unix_timestamp(concat( '2014 ', substring
(created_at,5,15)), 'yyyy MMM dd hh:mm:ss')) as timestamp) ts,
  text,
  tuser.time_zone

 FROM Mytweets_raw
 ;

 CREATE VIEW tweets_clean AS
 SELECT
   id,
   ts,
   text,
   m.country
  FROM tweets_simple t LEFT OUTER JOIN time_zone_map m ON t.time_zone =
 m.time_zone;


  -- Compute sentiment
 create view l1 as select id, words from Mytweets_raw lateral view explode
 (sentences(lower(text))) dummy as words;
 create view l2 as select id, word from l1 lateral view explode( words ) dummy as
 word ;


 create view l3 as select
     id,
     l2.word,
     case d.polarity
       when  'negative' then -1
       when 'positive' then 1
       else 0 end as polarity
  from l2 left outer join dictionary d on l2.word = d.word;
```

SQL ▾ Tab Width: 8 ▾ Ln 112, Col 2 ▾ INS

```
create table tweets_sentiment as select
  id,
  case
    when sum( polarity ) > 0 then 'positive'
    when sum( polarity ) < 0 then 'negative'
    else 'neutral' end as sentiment
 from l3 group by id;


 -- put everything back together and re-name sentiments...
CREATE TABLE tweetsbi
AS
SELECT
  t.*,
  s.sentiment
FROM tweets_clean t LEFT OUTER JOIN tweets_sentiment s on t.id = s.id;

-- data with tweet counts.....
CREATE TABLE tweetsbiaggr
AS
SELECT
  country,sentiment, count(sentiment) as tweet_count
FROM tweetsbi
group by country,sentiment;

-- store data for analysis......

CREATE VIEW A as select country,tweet_count as positive_response from
tweetsbiaggr where sentiment='positive';
CREATE VIEW B as select country,tweet_count as negative_response from
tweetsbiaggr where sentiment='negative';
CREATE VIEW C as select country,tweet_count as neutral_response from
tweetsbiaggr where sentiment='neutral';
CREATE TABLE tweetcompare as select A.*,B.negative_response as
negative_response,C.neutral_response as neutral_response from A join B on
A.country= B.country join C on B.country=C.country;

 -- permission to show data in Excel sheet for analysis ....
--grant SELECT ON TABLE tweetcompare to user jdk;
grant SELECT ON TABLE tweetcompare to user root;



-- for Tableau or Excel
-- UDAF sentiscore = sum(sentiment)*50  / count(sentiment)

-- context n-gram made readable
```

30. Now run the tweets.sql file using the hive command.

```
jdk@ubuntu:~$ hive --f /flumedir/tweets.sql
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.4.1.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4
j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-com
mon-2.0.1.jar!/hive-log4j2.properties
```

31. After running the script, we receive this output as SUCCESS.

```
jdk@ubuntu: ~
2016-07-24 13:58:11     End of local task; Time Taken: 2.921 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1469348690564_0004, Tracking URL = http://ubuntu:8088/proxy/a
pplication_1469348690564_0004/
Kill Command = /usr/local/hadoop/bin/hadoop job  -kill job_1469348690564_0004
Hadoop job information for Stage-5: number of mappers: 1; number of reducers: 0
2016-07-24 13:58:25,732 Stage-5 map = 0%,  reduce = 0%
2016-07-24 13:58:35,676 Stage-5 map = 100%,  reduce = 0%, Cumulative CPU 2.77 se
c
MapReduce Total cumulative CPU time: 2 seconds 770 msec
Ended Job = job_1469348690564_0004
Moving data to: hdfs://localhost:9000/user/hive/warehouse/tweetcompare
MapReduce Jobs Launched:
Stage-Stage-5: Map: 1   Cumulative CPU: 2.77 sec   HDFS Read: 7452 HDFS Write: 4
6 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 770 msec
OK
Time taken: 38.867 seconds
OK
Time taken: 0.332 seconds
jdk@ubuntu:~$
```

32. Now we look into all the created tables in the hive shell and default database.

```
hive> show tables;
OK
a
b
c
dictionary
l1
l2
l3
mytweets_raw
time_zone_map
tweetcompare
tweets_clean
tweets_sentiment
tweets_simple
tweetsbi
tweetsbiaggr
Time taken: 1.676 seconds, Fetched: 15 row(s)
hive>
```

33. The outputs can be seen as follows:

Tweets_simple

```
jdk@ubuntu: ~
 front of Virat Kohli. #IndVsWI https://t.co/kHFsmTB8GE NULL
757113923866800129      2014-07-24 07:23:20     RT @TrollKejri: My Delhi Guy Vir
at Kohli 200 run , @narendramodi Gujarat Guy 16 run. Delhi defeats Gujarat from
184 runs #ImandarSarkar NULL
757113930502311936      2014-07-24 07:23:22     RT @bhogleharsha: The first doub
le has taken time coming but there is little doubt there are many more on the wa
y #Kohli        NULL
757113931659751424      2014-07-24 07:23:22     RT @TrendieIN: "Trending India 0
1:21 AM IST"\n1. #Munich\n2. #HappyBirthdaySuriya\n3. #BSPInsultsWomen\n4. #IfIW
asRajini\n5. Amit Mishra\n6. Vira…      NULL
757113989902008320      2014-07-24 07:23:36     V Kohli is the youngest\nvisitin
g captain to enforce f/\no in West Indies. Already the\nyoungest Indian to enfor
ce\nf/o (v Ban,2015).#WIvIND    NULL
757114012299427840      2014-07-24 07:23:41     Virat ! https://t.co/BVRbsdnf6cN
ULL
757114069723799552      2014-07-24 07:23:55     1st Test: Virat Kohli's India Ey
e Big Win After West Indies Follow-On https://t.co/5sl2S1xbSe   NULL
757114094528921601      2014-07-24 07:24:01     @raghavabhay @michaelvaughan joe
 root pisses on pants while he faces Australian pacers - Kohli conquered those p
acers   NULL
757114107594027008      2014-07-24 07:24:04     #Kohli makes #history with #doub
le #century #against #Windies\nTop Trend:#NawazTheVoiceOfKashmir https://t.co/Qe
skkk69PS        NULL
757114127055777792      2014-07-24 07:24:09     1st Test: Virat Kohli's India Ey
```

Tweets_clean

```
jdk@ubuntu: ~
757114158047498240      ["rt","cricfit","virat","kohli","is","a","perfect","amba
ssador","for","cricket","says","sourav","ganguly","wivind","https","t.co","dkgmi
nym9v"]
757114168159854592      ["rt","bhogleharsha","people","always","think","their","
era","was","better","but","couldn't","ask","for","more","with","batsmanship","fr
om","kohli","and","root","and","ab","and","warn"]
757114189248917504      ["rt","mihika","07","msd","is","champion","nvirat","is",
"champion","nraina","is","champion","nevery","player","is","champion","n","nthat
's","all","dear","cricket","fans","so","u","need","to"]
757114202330890240      ["cricket","virat","kohli","double","century","west","in
dies","22nd","july","2016","https","t.co","k1jy6gw6rg","https","t.co","ty1jytyus
n"]
757114258522013696      ["rt","viratfanclub18","my","inspiration","virat","kohli
"]
757114258522013696      ["imvkohli","nall","the","best","team","india","https","
t.co","7y936fodjf"]
757114262322044928      ["rt","cricketopiacom","kohli's","12","test","hundreds",
"n","n5","in","australia","n3","in","india","n1","each","in","new","zealand","so
uth","africa","sri","lanka","amp","west","indies","https"]
757114294387482624      ["virat","kohli","scored","1096","runs","at","64.47","av
erage","as","a","captain","in","tests","https","t.co","i7tuldiooe"]
757114323143651328      ["virat","kohli","scored","1096","runs","at","64.47","av
erage","as","a","captain","in","tests","https","t.co","qcno44tjq0"]
Time taken: 31.912 seconds, Fetched: 34 row(s)
```