

APACHE FLUME

1934028

PRATIBA KR

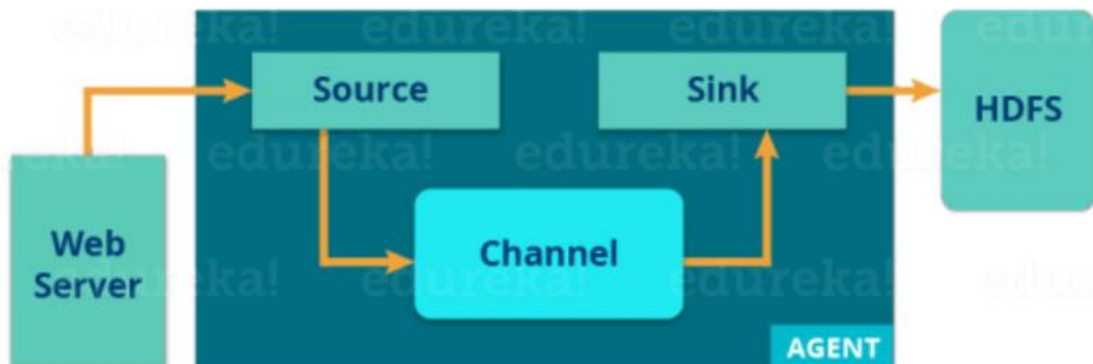
1934029

PREETHI S

INTRODUCTION

Apache Flume is a tool for data ingestion in HDFS. It collects, aggregates and transports large amount of streaming data such as log files, events from various sources like network traffic, social media, email messages etc. to HDFS. Flume is a highly reliable & distributed.

The main idea behind the Flume's design is to capture streaming data from various web servers to HDFS. It has simple and flexible architecture based on streaming data flows. It is fault-tolerant and provides reliability mechanism for Fault tolerance & failure recovery.



ADVANTAGES

- Flume is scalable, reliable, fault tolerant and customizable for different sources and sinks.
- Apache Flume can store data in centralized stores (i.e data is supplied from a single store) like HBase & HDFS.
- Flume is horizontally scalable.
- If the read rate exceeds the write rate, Flume provides a steady flow of data between read and write operations.
- Flume provides reliable message delivery. The transactions in Flume are channel-based where two transactions (one sender & one receiver) are maintained for each message.
- Using Flume, we can ingest data from multiple servers into Hadoop.

- It gives us a solution which is reliable and distributed and helps us in collecting, aggregating and moving large amount of data sets like Facebook, Twitter and e-commerce websites.
- It helps us to ingest online streaming data from various sources like network traffic, social media, email messages, log files etc. in HDFS.
- It supports a large set of sources and destinations types.

ARCHITECTURE

The flume agent has 3 components: source, sink and channel.

1. Source: It accepts the data from the incoming streamline and stores the data in the channel.
2. Channel: In general, the reading speed is faster than the writing speed. Thus, we need some buffer to match the read & write speed difference. Basically, the buffer acts as a intermediary storage that stores the data being transferred temporarily and therefore prevents data loss. Similarly, channel acts as the local storage or a temporary storage between the source of data and persistent data in the HDFS.
3. Sink: Then, our last component i.e. Sink, collects the data from the channel and commits or writes the data in the HDFS permanently.

PREREQUISITE FOR INSTALLATION

- Apache Hadoop
- Java

INSTALLATION

1. First we have to download Apache Flume 1.9.0.
2. Locate the tar file that you have downloaded.
3. Extract the tar file using the below command:
4. `tar xzf apache-flume-1.9.0-bin.tar.gz`

```
hdoopp@preethi:/home/preethi/flume$ cd '/home/preethi/flume'
hdoopp@preethi:/home/preethi/flume$ tar xzf apache-flume-1.9.0-bin.tar.gz
tar: apache-flume-1.9.0-bin/conf: Cannot utime: Operation not permitted
tar: apache-flume-1.9.0-bin/bin: Cannot utime: Operation not permitted
tar: apache-flume-1.9.0-bin/doap_Flume.rdf: Cannot open: File exists
tar: apache-flume-1.9.0-bin/README.md: Cannot open: File exists
tar: apache-flume-1.9.0-bin/DEVNOTES: Cannot open: File exists
tar: apache-flume-1.9.0-bin/conf/log4j.properties: Cannot open: File exists
tar: apache-flume-1.9.0-bin/conf/flume-env.sh.template: Cannot open: File exists
tar: apache-flume-1.9.0-bin/conf/flume-conf.properties.template: Cannot open: File exists
tar: apache-flume-1.9.0-bin/conf/flume-env.ps1.template: Cannot open: File exists
tar: apache-flume-1.9.0-bin/NOTICE: Cannot open: File exists
tar: apache-flume-1.9.0-bin/CHANGELOG: Cannot open: File exists
tar: apache-flume-1.9.0-bin/bin/flume-ng: Cannot open: File exists
tar: apache-flume-1.9.0-bin/bin/flume-ng.ps1: Cannot open: File exists
tar: apache-flume-1.9.0-bin/bin/flume-ng.cmd: Cannot open: File exists
tar: apache-flume-1.9.0-bin/RELEASE-NOTES: Cannot open: File exists
```

5. Now we have to set the FLUME_HOME path in the .bashrc file. For this open .bashrc file in nano editor.

Add below parameters in the .bashrc file.

```
export FLUME_HOME=/home/dataflair/apache-flume-1.9.0-bin
```

```
export PATH=$PATH:$FLUME_HOME/bin
```

6. Refresh the .bashrc file by using the below command:

```
source .bashrc
```

7. The flume has now successfully been installed on our system. To verify Flume installation use below command:

```
flume-ng version
```

```
hdoopp@preethi:/home/preethi/flume$ flume-ng version
/home/hdoopp/hadoop/libexec/hadoop-functions.sh: line 2366: HADOOP_ORG.APACHE.FLUME.TOOLS.GETJAVAPROPERTY_USER: invalid variable name
/home/hdoopp/hadoop/libexec/hadoop-functions.sh: line 2461: HADOOP_ORG.APACHE.FLUME.TOOLS.GETJAVAPROPERTY_OPTS: invalid variable name
Flume 1.9.0
Source code repository: https://git-wip-us.apache.org/repos/asf/flume.git
Revision: d4fcab4f501d41597bc616921329a4339f73585e
Compiled by fszabo on Mon Dec 17 20:45:25 CET 2018
From source with checksum 35db629a3bda49d23e9b3690c80737f9
hdoopp@preethi:/home/preethi/flume$
```

We have successfully installed Apache Flume 1.9.0 on Ubuntu.

CONFIGURATION

1. Locate the access log file in your local file system which you want to move to HDFS.
2. Now, we have to create a file named flume.conf inside the /home/dataflair/apache-flume-1.6.0-bin/conf directory. Use the below-mentioned command for creating the file “flume.conf”:

```
touch flume.conf
```

3. Now we will start the Flume for copying files using the below command:

```
bin/flume-ng agent -conf ./conf/ -f conf/flume.conf -n agent1 -Dflume.root.logger=DEBUG
```

4. We can check whether data gets copied in HDFS or not by either using a web console (<http://localhost:9870>) or view the files in HDFS by using the command prompt. On the web console you can view your files in the “/flume01” directory.

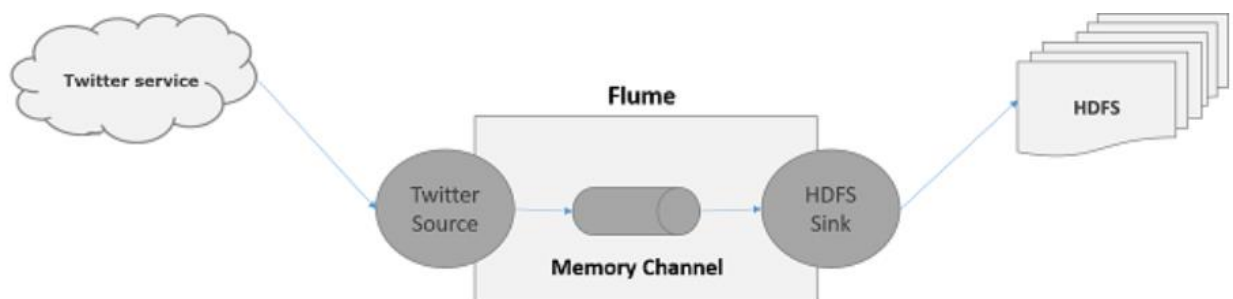
We have successfully configured Apache Flume 1.9.0 on ubuntu

STREAMING TWITTER DATA

Using Flume, we can fetch data from various services and transport it to centralized stores (HDFS and HBase). This section explains how to fetch data from Twitter service and store it in HDFS using Apache Flume.

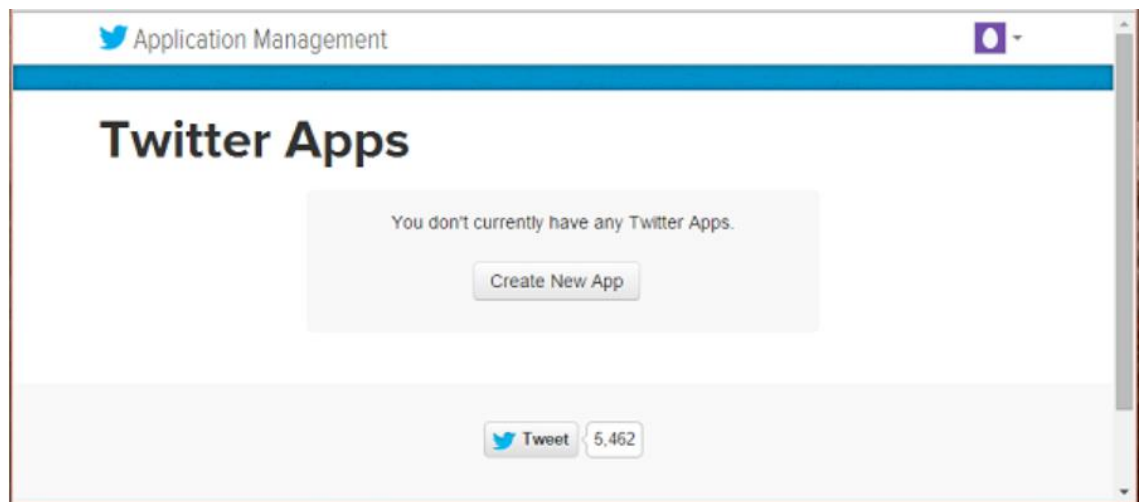
As discussed in Flume Architecture, a webserver generates log data and this data is collected by an agent in Flume. The channel buffers this data to a sink, which finally pushes it to centralized stores.

we will create an application and get the tweets from it using the experimental twitter source provided by Apache Flume. We will use the memory channel to buffer these tweets and HDFS sink to push these tweets into the HDFS.



To fetch Twitter data, we will have to follow the steps given below –

- Create a twitter Application
 - Install / Start HDFS
 - Configure Flume
-
- **CREATING A TWITTER DEVELOPER ACCOUNT**
 1. To create a Twitter application, click on the following link <https://apps.twitter.com/>. Sign in to your Twitter account. You will have a Twitter Application Management window where you can create, delete, and manage Twitter Apps.



2. Click on the Create New App button. You will be redirected to a window where you will get an application form in which you have to fill in your details in order to create the App.

Create an application

Application Details
Name *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *


Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

3. Fill in the details, accept the Developer Agreement when finished, click on the Create your Twitter application button which is at the bottom of the page. If everything goes fine, an App will be created with the given details as shown below.

Application Management

Tp_Flume_Example [Test OAuth](#)

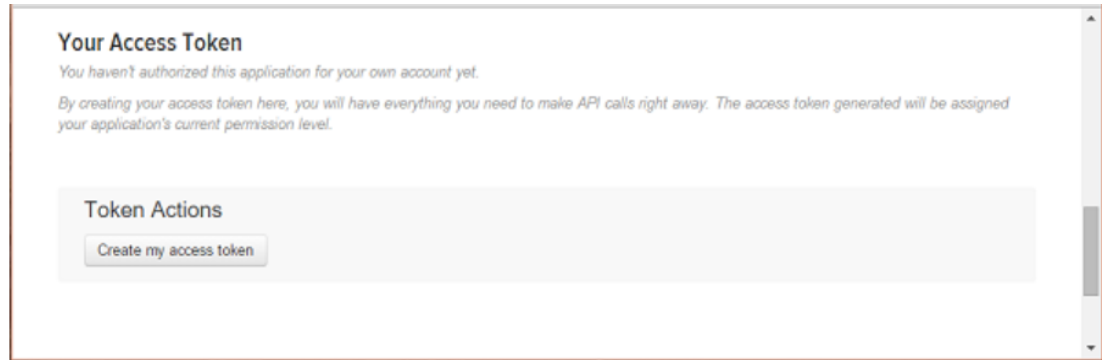
[Details](#) [Settings](#) [Keys and Access Tokens](#) [Permissions](#)

 **fetch twitter data**
<http://tpflumeexample.com>

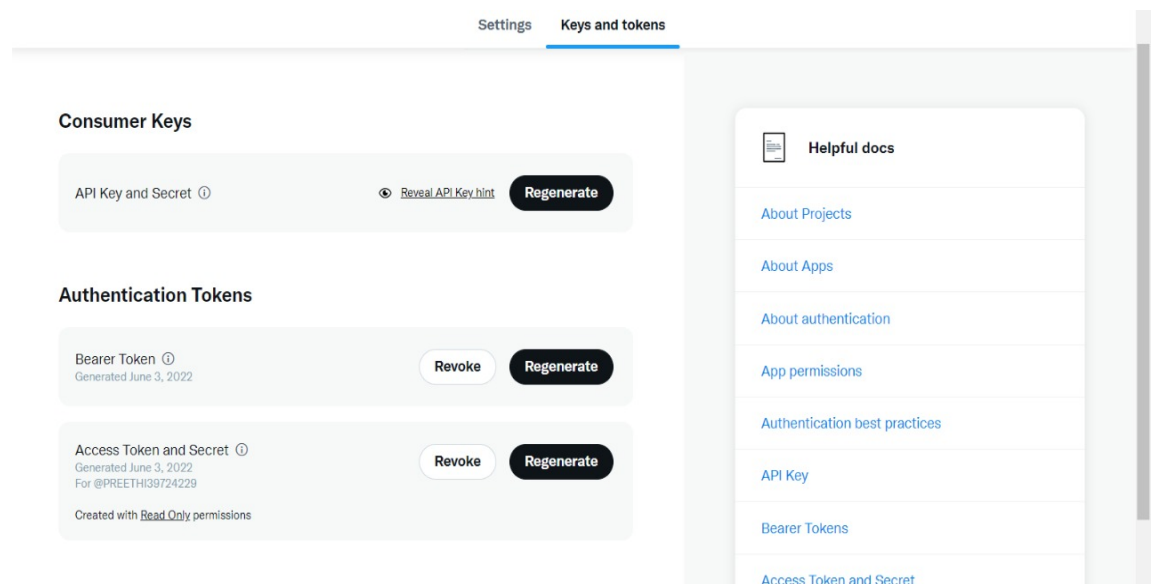
Organization
Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

4. Under keys and Access Tokens tab at the bottom of the page, you can observe a button named Create my access token. Click on it to generate the access token.



5. Finally, click on the Test OAuth button which is on the right side top of the page. This will lead to a page which displays your Consumer key, Consumer secret, Access token, and Access token secret. Copy these details. These are useful to configure the agent in Flume.



- **STARTING HDFS**

1. Install Hadoop. If Hadoop is already installed in your system, verify the installation using Hadoop version command, as shown below.

```
$ hadoop version
```

2. Browse through the sbin directory of Hadoop and start yarn and Hadoop dfs (distributed file system) as shown below.

```

hdoopp@preethi:~$ cd hadoop
hdoopp@preethi:~/hadoop$ cd sbin
hdoopp@preethi:~/hadoop/sbin$ ./start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hdoopp@preethi:~/hadoop/sbin$ jps
9328 Application
11079 NodeManager
10815 ResourceManager
11247 Jps
hdoopp@preethi:~/hadoop/sbin$ ./start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [preethi]
hdoopp@preethi:~/hadoop/sbin$ jps
9328 Application
12070 SecondaryNameNode
11079 NodeManager
11739 NameNode
11868 DataNode
10815 ResourceManager
12239 Jps
hdoopp@preethi:~/hadoop/sbin$ cd

```

3. In Hadoop DFS, you can create directories using the command `mkdir`. Browse through it and create a directory with the name `twitter_data` in the required path as shown below.

```
$cd /$Hadoop_Home/bin/
```

```
$ hdfs dfs -mkdir hdfs://localhost:9000/user/Hadoop/twitter_data
```

- **CONFIGURE FLUME**

We have to configure the source, the channel, and the sink using the configuration file in the `conf` folder.

1. Set the classpath variable to the `lib` folder of Flume in `Flume-env.sh` file as shown below.

```
export CLASSPATH=$CLASSPATH:/FLUME_HOME/lib/*
```

2. Configuration

To configure source, you have to provide values to the following properties :

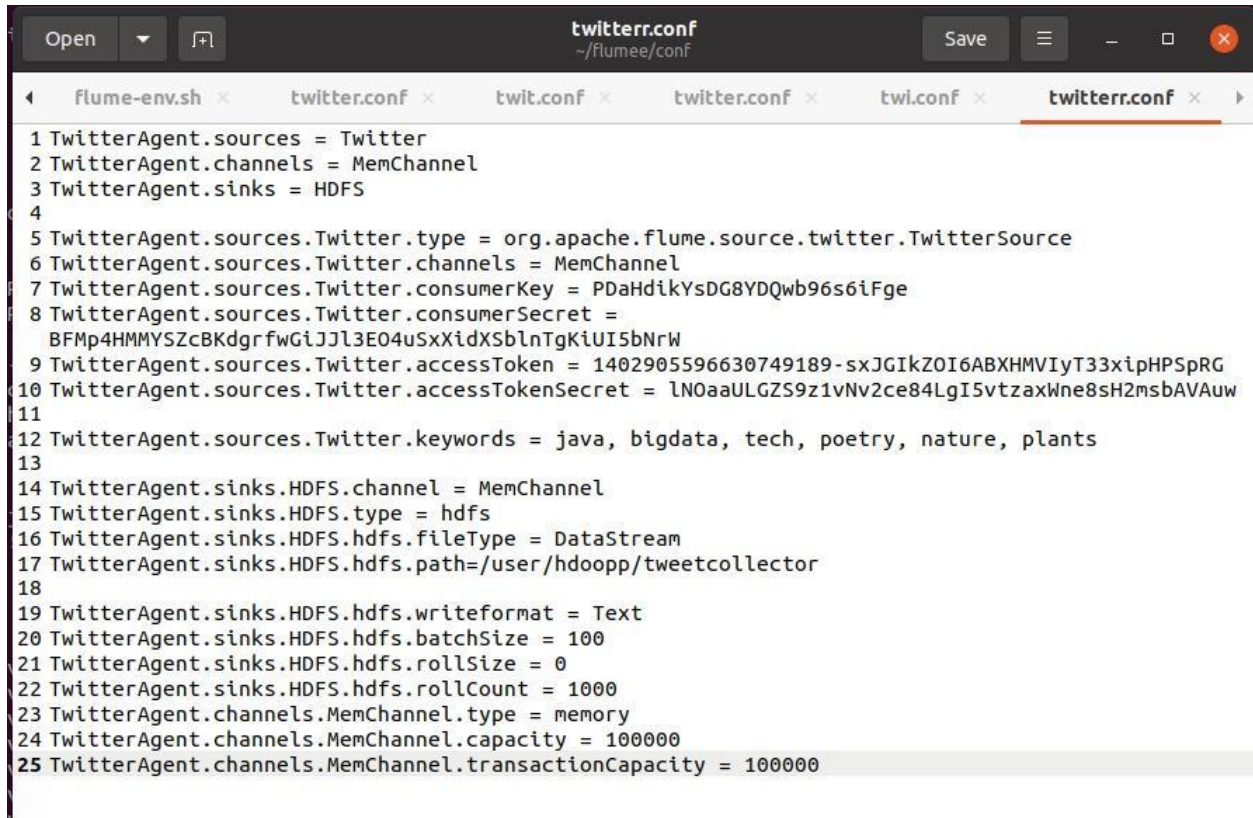
- Channels
- Source type : org.apache.flume.source.twitter.TwitterSource
- consumerKey – The OAuth consumer key
- consumerSecret – OAuth consumer secret
- accessToken – OAuth access token
- accessTokenSecret – OAuth token secret
- batchSize – Maximum number of twitter messages that should be in a twitter batch. The default value is 1000 (optional).
- batchSizeInMillis – Maximum number of milliseconds to wait before closing a batch. The default value is 1000 (optional).

To configure channel, you have to provide values to the following properties :

- type – It holds the type of the channel. In our example, the type is MemChannel.
- Capacity – It is the maximum number of events stored in the channel. Its default value is 100 (optional).
- TransactionCapacity – It is the maximum number of events the channel accepts or sends. Its default value is 100 (optional).

To configure sink, you have to provide values to the following properties :

- Channel
- type – hdfs
- hdfs.path – the path of the directory in HDFS where data is to be stored.

A screenshot of a text editor window titled 'twitter.conf' with the path '~/.flume/conf'. The editor shows a configuration file for a Twitter agent. The tabs at the top include 'flume-env.sh', 'twitter.conf', 'twit.conf', 'twitter.conf', 'twi.conf', and 'twitter.conf'. The configuration lines are as follows:

```
1 TwitterAgent.sources = Twitter
2 TwitterAgent.channels = MemChannel
3 TwitterAgent.sinks = HDFS
4
5 TwitterAgent.sources.Twitter.type = org.apache.flume.source.twitter.TwitterSource
6 TwitterAgent.sources.Twitter.channels = MemChannel
7 TwitterAgent.sources.Twitter.consumerKey = PDaHdikYsDG8YDQwb96s6iFge
8 TwitterAgent.sources.Twitter.consumerSecret =
  BFMP4HMMYSZcBKdgrfwGiJJl3E04uSxXidXSblnTgKiUI5bNrW
9 TwitterAgent.sources.Twitter.accessToken = 1402905596630749189-sxJGikZOI6ABXHMVIyT33xiPHSPRG
10 TwitterAgent.sources.Twitter.accessTokenSecret = lNOaaULGZS9z1vNv2ce84LgI5vtzaxWne8sH2msbAVAUw
11
12 TwitterAgent.sources.Twitter.keywords = java, bigdata, tech, poetry, nature, plants
13
14 TwitterAgent.sinks.HDFS.channel = MemChannel
15 TwitterAgent.sinks.HDFS.type = hdfs
16 TwitterAgent.sinks.HDFS.hdfs.fileType = DataStream
17 TwitterAgent.sinks.HDFS.hdfs.path=/user/hdoopp/tweetcollector
18
19 TwitterAgent.sinks.HDFS.hdfs.writeformat = Text
20 TwitterAgent.sinks.HDFS.hdfs.batchSize = 100
21 TwitterAgent.sinks.HDFS.hdfs.rollSize = 0
22 TwitterAgent.sinks.HDFS.hdfs.rollCount = 1000
23 TwitterAgent.channels.MemChannel.type = memory
24 TwitterAgent.channels.MemChannel.capacity = 100000
25 TwitterAgent.channels.MemChannel.transactionCapacity = 100000
```

3. Browse through the Flume home directory and execute the application as shown below.

```
$ cd $FLUME_HOME
```

```
$ bin/flume-ng agent --conf ./conf/ -f conf/twitter.conf
```

```
Dflume.root.logger=DEBUG,console -n TwitterAgent
```


[illegible]

[illegible]

4. You can access the Hadoop Administration Web UI using the URL given below.

← → ↻ localhost:9870/explorer.html/

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Browse Directory

Show 25 entries Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Mar 16 10:38	0	0 B	Folder	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 06 19:25	0	0 B	Temperature	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 07 14:42	0	0 B	Temperaturee	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Jun 03 20:08	0	0 B	flume_dat	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Jun 03 20:05	0	0 B	flume_data	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 25 10:06	0	0 B	marks	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 24 18:41	0	0 B	matrix	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 24 11:22	0	0 B	multi	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 24 17:11	0	0 B	production	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 24 10:02	0	0 B	stopwordex	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	hdoopp	supergroup	0 B	Apr 12 19:27	0	0 B	temperature	<input type="checkbox"/>