Dāvis Sporāns

DS16042

**DatZ6082: Lielo datu tehnoloģijas**

2. Laboratorijas darbs – Flume, Solr lietošana

Lai izprastu Flume pielietojumu, tika izvēlēta Twitter datu plūsma, jo tīmeklī pieejami daudz paskaidrojumi un pamācības tieši ar twitter datu plūsmu.

1. Nepieciešams piešķirt tiesības Flume izgūt twitter datus, tāpēc reģistrējos saitē <http://apps.twitter.com> un ieguvu piekļuves atslēgas, kuras pēc tam būs nepieciešamas konfigurējot Flume.
2. Flume darbina atseviški, lai pārbaudītu tā konfigurāciju. Tāpēc izveido mapi HDFS ‘twitter’ un apakšmapi ‘feed’.
3. Tālāk izveido konfigurācijas failu ar tā saturu:

# Naming the components on the current agent.

TwitterAgent.sources = Twitter

TwitterAgent.channels = MemChannel

TwitterAgent.sinks = HDFS

# Describing/Configuring the source

#TwitterAgent.sources.Twitter.type = com.cloudera.flume.source.TwitterSource

TwitterAgent.sources.Twitter.type = org.apache.flume.source.twitter.TwitterSource

TwitterAgent.sources.Twitter.consumerKey = \*masked\*

TwitterAgent.sources.Twitter.consumerSecret = \*masked\*

TwitterAgent.sources.Twitter.accessToken = \*masked\*

TwitterAgent.sources.Twitter.accessTokenSecret = \*masked\*

TwitterAgent.sources.Twitter.keywords = #LV100, #SundayMorning

# Describing/Configuring the sink

TwitterAgent.sinks.HDFS.type = hdfs

TwitterAgent.sinks.HDFS.hdfs.path = /user/cloudera/twitter/feed

TwitterAgent.sinks.HDFS.hdfs.fileType = DataStream

TwitterAgent.sinks.HDFS.hdfs.writeFormat = Text

TwitterAgent.sinks.HDFS.hdfs.filePrefix = twitter-

TwitterAgent.sinks.HDFS.hdfs.rollInterval = 0

TwitterAgent.sinks.HDFS.hdfs.rollSize = 524288

TwitterAgent.sinks.HDFS.hdfs.rollCount = 0

TwitterAgent.sinks.HDFS.hdfs.idleTimeout = 0

TwitterAgent.sinks.HDFS.hdfs.batchSize = 100

TwitterAgent.sinks.HDFS.hdfs.threadsPoolSize = 2

TwitterAgent.sinks.HDFS.hdfs.round = true

TwitterAgent.sinks.HDFS.hdfs.roundUnit = hour

# Describing/Configuring the channel

TwitterAgent.channels.MemChannel.type = memory

TwitterAgent.channels.MemChannel.capacity = 10000

TwitterAgent.channels.MemChannel.transactionCapacity = 100

# Binding the source and sink to the channel

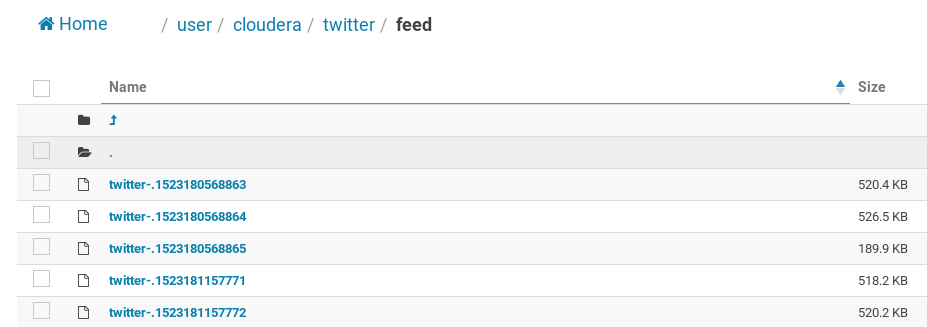
TwitterAgent.sources.Twitter.channels = MemChannel

TwitterAgent.sinks.HDFS.channel = MemChannel

1. Un darbina Flume ar komandu:

flume-ng agent --conf /home/cloudera/twitter\_flume --conf-file flume\_process\_twitter.conf Dflume.root.logger=DEBUG,console -n TwitterAgent

1. No tā ka HDFS tiek ģenerēti faili, var secināt, ka risinājums darbojas korekti.



1. Lai savienotu Flume ar Solr servisu, nepieciešmas mainīt config faila saturu, norādot HDFS sink vietā Solr sink:

# Naming the components on the current agent.

TwitterAgent.sources = Twitter

TwitterAgent.channels = MemChannel

TwitterAgent.sinks = HDFS

# Describing/Configuring the source

#TwitterAgent.sources.Twitter.type = com.cloudera.flume.source.TwitterSource

TwitterAgent.sources.Twitter.type = org.apache.flume.source.twitter.TwitterSource

TwitterAgent.sources.Twitter.consumerKey = ekk5WyiW6geBFaOQyThHlYfgN

TwitterAgent.sources.Twitter.consumerSecret = dfMgIASl3YKibhAkgz6dcKjdrdj8cIiuH9QdZzEVY3hmmpzhLN

TwitterAgent.sources.Twitter.accessToken = 44428046-hS7MDoaZYtfdll3bzjZV1vlVHpq2GbUw5cvqJn3it

TwitterAgent.sources.Twitter.accessTokenSecret = UUXdaXTHZfgtVqSJlX4rJpcQNFupMUsHVwyF7rkyMzAyB

TwitterAgent.sources.Twitter.keywords = #LV100, #MondayMotivation

# Describing/Configuring the sink

TwitterAgent.sinks.solrSink.type = org.apache.flume.sink.solr.morphline.MorphlineSolrSink

TwitterAgent.sinks.solrSink.morphlineFile = /home/cloudera/twitter\_flume/morphline.conf

TwitterAgent.sinks.solrSink.batchSize = 1000

TwitterAgent.sinks.solrSink.batchDurationMillis = 1000

TwitterAgent.sinks.solrSink.morphlineId = morphline1

# Describing/Configuring the channel

TwitterAgent.channels.MemChannel.type = memory

TwitterAgent.channels.MemChannel.capacity = 10000

TwitterAgent.channels.MemChannel.transactionCapacity = 100

# Binding the source and sink to the channel

TwitterAgent.sources.Twitter.channels = MemChannel

TwitterAgent.sinks.solrSink.channel = MemChannel

1. Papildus nepieciešams definēt morphlines.conf failu, kurš nepieciešams Solr konfigurācijai:

# Specify server locations in a SOLR\_LOCATOR variable; used later in variable substitutions:

SOLR\_LOCATOR : {

# Name of solr collection

collection : collection1

# ZooKeeper ensemble

zkHost : "$ZK\_HOST"

# Relative or absolute path to a directory containing conf/solrconfig.xml and conf/schema.xml

# If this path is uncommented it takes precedence over the configuration stored in ZooKeeper.

# solrHomeDir : "example/solr/collection1"

# The maximum number of documents to send to Solr per network batch (throughput knob)

# batchSize : 100

}

# Specify an array of one or more morphlines, each of which defines an ETL

# transformation chain. A morphline consists of one or more (potentially

# nested) commands. A morphline is a way to consume records (e.g. Flume events,

# HDFS files or blocks), turn them into a stream of records, and pipe the stream

# of records through a set of easily configurable transformations on it's way to

# Solr (or a MapReduceIndexerTool RecordWriter that feeds via a Reducer into Solr).

morphlines : [

{

# Name used to identify a morphline. E.g. used if there are multiple morphlines in a

# morphline config file

id : morphline1

# Import all morphline commands in these java packages and their subpackages.

# Other commands that may be present on the classpath are not visible to this morphline.

importCommands : ["org.kitesdk.\*\*", "org.apache.solr.\*\*"]

commands : [

{

# Parse Avro container file and emit a record for each avro object

readAvroContainer {

# Optionally, require the input record to match one of these MIME types:

# supportedMimeTypes : [avro/binary]

# Optionally, use a custom Avro schema in JSON format inline:

# schemaString : """<json can go here>"""

# Optionally, use a custom Avro schema file in JSON format:

# schemaFile : /path/to/syslog.avsc

}

}

{

extractAvroPaths {

flatten : false

paths : {

id : /id

text : /text

user\_friends\_count : /user\_friends\_count

user\_location : /user\_location

user\_description : /user\_description

user\_statuses\_count : /user\_statuses\_count

user\_followers\_count : /user\_followers\_count

user\_name : /user\_name

user\_screen\_name : /user\_screen\_name

created\_at : /created\_at

retweet\_count : /retweet\_count

retweeted : /retweeted

in\_reply\_to\_user\_id : /in\_reply\_to\_user\_id

source : /source

in\_reply\_to\_status\_id : /in\_reply\_to\_status\_id

media\_url\_https : /media\_url\_https

expanded\_url : /expanded\_url

}

}

}

# Consume the output record of the previous command and pipe another record downstream.

#

# convert timestamp field to native Solr timestamp format

# e.g. 2012-09-06T07:14:34Z to 2012-09-06T07:14:34.000Z

{

convertTimestamp {

field : created\_at

inputFormats : ["yyyy-MM-dd'T'HH:mm:ss'Z'", "yyyy-MM-dd"]

inputTimezone : America/Los\_Angeles

# outputFormat : "yyyy-MM-dd'T'HH:mm:ss.SSSZ"

outputTimezone : UTC

}

}

{

sanitizeUnknownSolrFields {

# Location from which to fetch Solr schema

solrLocator : ${SOLR\_LOCATOR}

# renameToPrefix : "ignored\_"

}

}

# log the record at DEBUG level to SLF4J

{ logDebug { format : "output record: {}", args : ["@{}"] } }

# load the record into a SolrServer or MapReduce SolrOutputFormat.

{

loadSolr {

solrLocator : ${SOLR\_LOCATOR}

}

}

]

}

]

1. Tāpat kā iepriekš flume darbina ar komandu:

flume-ng agent -c --conf /home/cloudera/twitter\_flume --conf-file flume\_process\_twitter.conf Dflume.root.logger=DEBUG,console -n TwitterAgent

1. Tiek iegūti twitter dati un indeksējot tiek noglabāti Solr serverī.

