Flume+Kafka+SparkStreaming

测试文档

目录

一、Flume 连通 Kafka 配置

```
a1.sources = r1
a1.channels = c1
a1.sinks =s1

#sources 端配置
a1.sources.r1.type=exec
a1.sources.r1.command=tail -F /usr/local/soft/flume/flume_dir/kafka.log
a1.sources.r1.channels=c1

#channels 端配置
a1.channels.c1.type=memory
a1.channels.c1.capacity=10000
a1.channels.c1.transactionCapacity=100

#设置 Kafka 接收器
a1.sinks.s1.type= org.apache.flume.sink.kafka.KafkaSink

#设置 Kafka 的 broker 地址和端口
a1.sinks.s1.brokerList=manager:9092,namenode:9092,datanode:9092

#设置 Kafka 的 Topic
a1.sinks.s1.topic=realtime

#设置序列化方式
a1.sinks.s1.serializer.class=kafka.serializer.StringEncoder
a1.sinks.s1.channel=c1
```

注意，关于配置文件中注意 3 点：

1.配置文件：

a. a1.sources.r1.command=tail -F /usr/local/soft/flume/flume_dir//kafka.log

b. a1.sinks.s1.brokerList= manager:9092,namenode:9092,datanode:9092

c . a1.sinks.s1.topic=realtime

2.很明显，由配置文件可以了解到:

a.我们需要在/usr/local/soft/flume/flume_dir/下建一个 kafka.log 的文件，且向文件中输出内容（下面会说到）；

b.flume 连接到 kafka 的地址是 manager:9092,namenode:9092,datanode:9092，注意不要配置出错了；

c.flume 会将采集后的内容输出到Kafka topic 为realtime 上，所以我们启动 zk,kafka 后需要打开一个终端消费 topic realtime 的内容。这样就可以看到 flume 与 kafka 之间玩起来了~~

二、编写测试脚本 kafka_output.sh

a. 在/usr/local/soft/flume/flume_dir/下建立空文件 kafka.log。在 root 用户目录下新建脚本 kafka_output.sh(一定要给予可执行权限)，用来向 kafka.log 输入内容，脚本内容如下:

```
for((i=0;i<=1000;i++));
do echo "kafka_test-"+$i>>/usr/local/soft/flume/flume_dir/kafka.log;
done
```

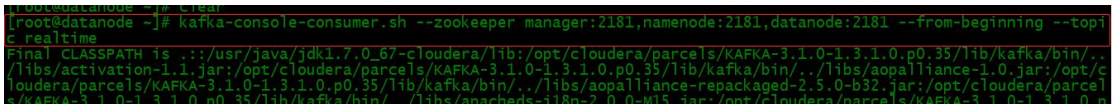
b. 在 Cloudera Manger(CM)上启动 Zookeeper,Kafka

c. 在 Kafka 集群上创建主题 realtime：

```
kafka-topics.sh  --create --zookeeper manager:2181,namenode:2181,datanode:2181 --replication-factor 3 --partitions 1  --topic  realtime
```

d. 打开新终端，在 kafka 安装目录下执行如下命令，生成对 topic realtime 的消费：

```
kafka-console-consumer.sh  --zookeeper  manager:2181,namenode:2181,datanode:2181 --from-beginning --topic realtime
```



e. 启动 Flume(CM 上启动也行)：

1）cd /opt/cloudera/parcels/CDH-5.7.5-1.cdh5.7.5.p0.3

2) bin/flume-ng agent --conf conf --conf-file etc/flume-ng/conf.empty/flume-conf.properties --name a1 -Dflume.root.logger=INFO,console

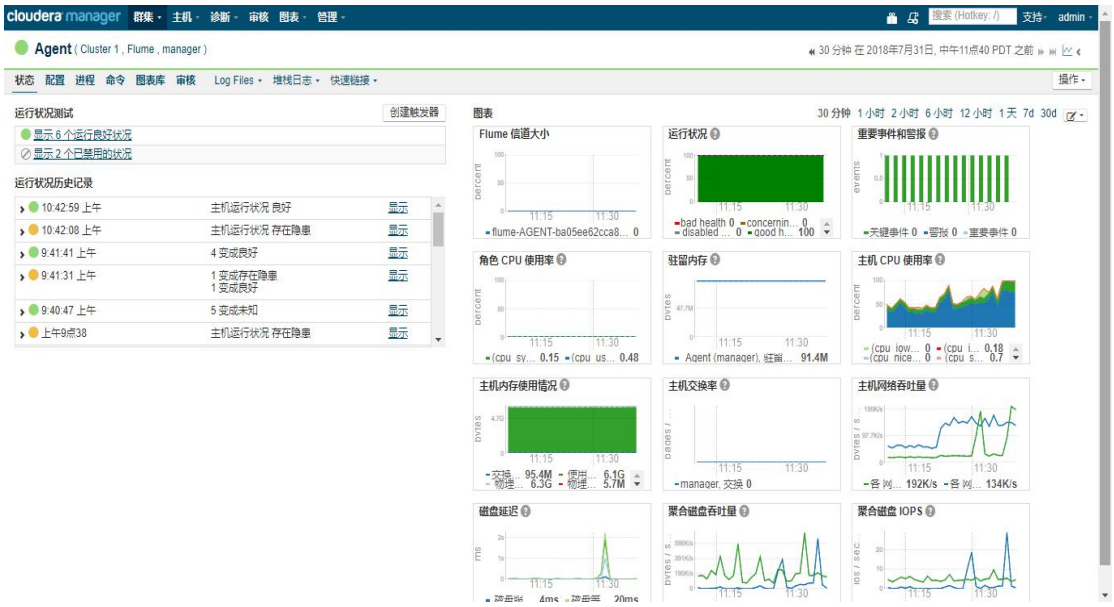
```
[root@manager ~]# cd /opt/cloudera/parcels/CDH-5.7.5-1.cdh5.7.5.p0.3
[root@manager CDH-5.7.5-1.cdh5.7.5.p0.3]# bin/flume-ng agent --conf conf --conf-file etc/flume-ng/conf.empty/kafka.properties --name agent -Dflume.root.logger=INFO,console
```

f. 执行 kafka_output.sh 脚本（注意观察 kafka.log 内容及消费终端接收到的内容）

```
[root@manager ~]# ./kafka_output.sh
```

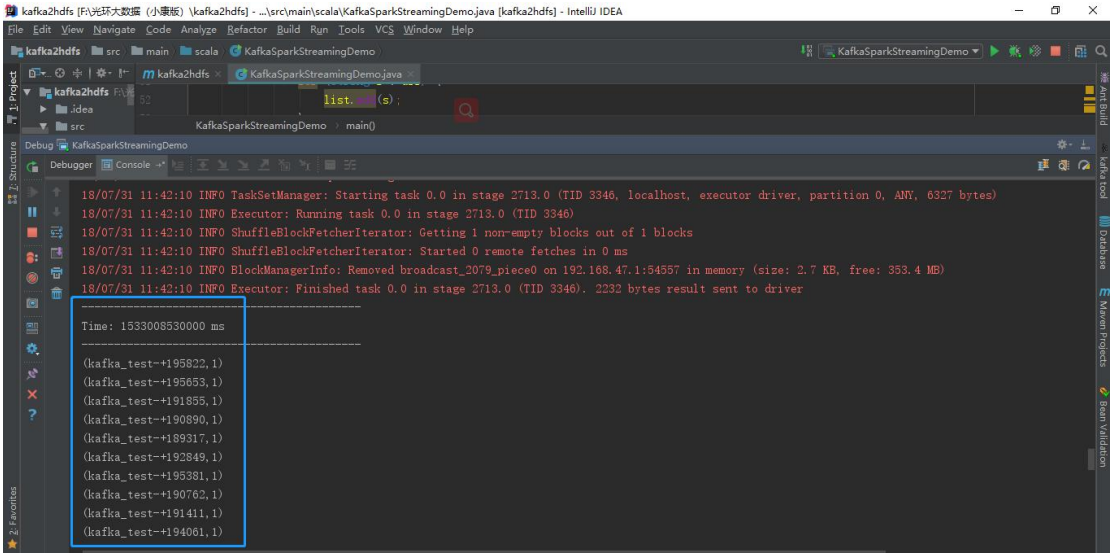
Flume 监控端：

```
18/07/31 11:36:27 INFO node.Application: Starting Channel c1
18/07/31 11:36:28 INFO instrumentation.MonitoredCounterGroup: Monitored counter group for type: CHANNEL, name: c1: Succes
sfully registered new MBean.
18/07/31 11:36:28 INFO instrumentation.MonitoredCounterGroup: Component type: CHANNEL, name: c1 started
18/07/31 11:36:28 INFO node.Application: Starting Sink k1
18/07/31 11:36:28 INFO node.Application: Starting Source s1
18/07/31 11:36:28 INFO source.ExecSource: Exec source starting with command:tail -F /usr/local/soft/flume/flume_dir/kafka
.log
18/07/31 11:36:28 INFO instrumentation.MonitoredCounterGroup: Monitored counter group for type: SOURCE, name: s1: Success
fully registered new MBean.
18/07/31 11:36:28 INFO instrumentation.MonitoredCounterGroup: Component type: SOURCE, name: s1 started
18/07/31 11:36:50 INFO utils.VerifiableProperties: Verifying properties
18/07/31 11:36:52 INFO utils.VerifiableProperties: Property key.serializer.class is overridden to kafka.serializer.String
Encoder
18/07/31 11:36:52 INFO utils.VerifiableProperties: Property metadata.broker.list is overridden to manager:9092,namenode:9
092,datanode:9092
18/07/31 11:36:52 INFO utils.VerifiableProperties: Property request.required.acks is overridden to 1
18/07/31 11:36:52 INFO utils.VerifiableProperties: Property serializer.class is overridden to kafka.serializer.DefaultEnc
oder
18/07/31 11:36:56 INFO instrumentation.MonitoredCounterGroup: Monitored counter group for type: SINK, name: k1: Successfu
lly registered new MBean.
18/07/31 11:36:56 INFO instrumentation.MonitoredCounterGroup: Component type: SINK, name: k1 started
18/07/31 11:36:57 INFO client.ClientUtils$: Fetching metadata from broker BrokerEndPoint(2,datanode,9092) with correlatio
n id 0 for 1 topic(s) Set(realtime)
18/07/31 11:36:58 INFO producer.SyncProducer: Connected to datanode:9092 for producing
18/07/31 11:36:58 INFO producer.SyncProducer: Disconnecting from datanode:9092
18/07/31 11:37:00 INFO producer.SyncProducer: Connected to manager:9092 for producing
```



Kafka 消费端：

Sparkstreaming 日志处理端：

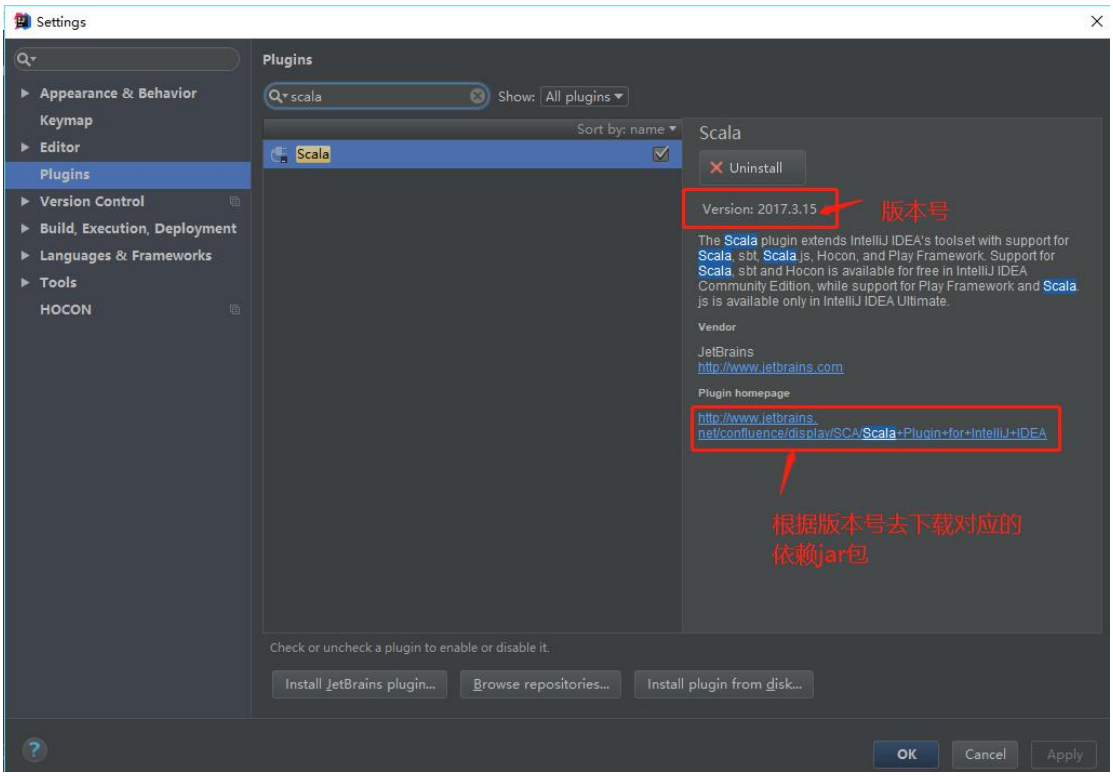


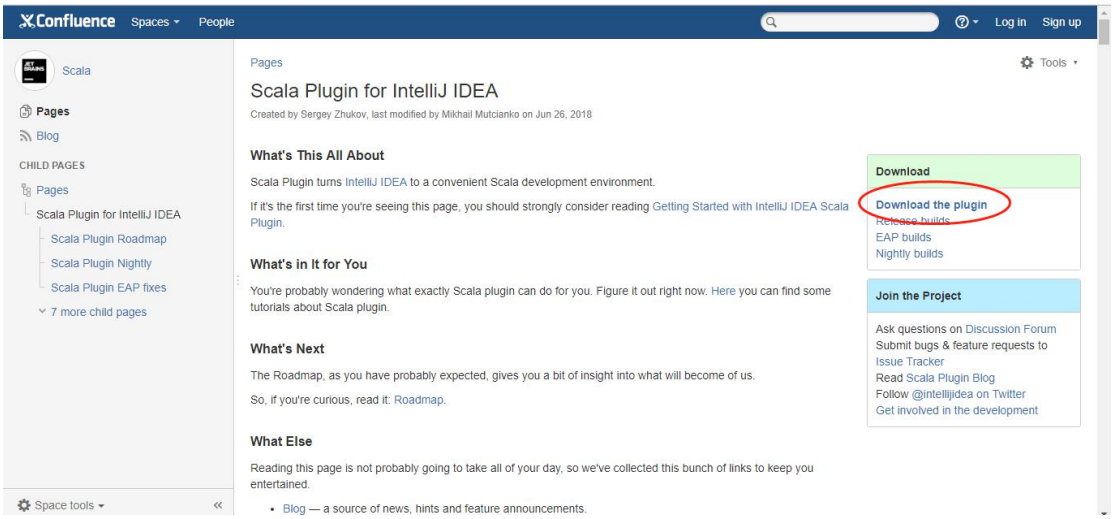
三、Sparkstreaming 代码

```
import org.apache.kafka.common.serialization.StringDeserializer
import org.apache.spark.SparkConf
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.streaming.kafka010._
import org.apache.spark.streaming.kafka010.LocationStrategies.PreferConsistent
import org.apache.spark.streaming.kafka010.ConsumerStrategies.Subscribe


/**
  * @author: create by LuJuHui
```

```
 * @ date:2018/7/31
 */
object Kafka2Scala2WC {
  def main(args: Array[String]): Unit = {
    val conf = new SparkConf().setAppName("Kafka2Scala2WC").setMaster("local[*]")
    val ssc = new StreamingContext(conf, Seconds(5))
    val kafkaParams = Map[String, Object](
      "bootstrap.servers" -> "manager:9092,namenode:9092,datanode:9092",
      "key.deserializer" -> classOf[StringDeserializer],
      "value.deserializer" -> classOf[StringDeserializer],
      "group.id" -> "kafka_wc",
      "auto.offset.reset" -> "latest",
      "enable.auto.commit" -> (false: java.lang.Boolean)
    )

    val topics = Array("realtime")
    val data = KafkaUtils.createDirectStream[String, String](
      ssc,
      PreferConsistent,
      Subscribe[String, String](topics, kafkaParams)
    )
    val lines = data.map(_.value())
    val words = lines.flatMap(_.split(" "))
    val wordAndOne = words.map((_, 1))
    val reduced = wordAndOne.reduceByKey(_ + _)
    reduced.print()

    ssc.start()
    ssc.awaitTermination()

  }

}
```

【注意：在新建 maven 项目是一定要根据 scala 版本去添加对应的依赖 jar 包，否则会报错】





2018.1.8	181—182	Mar 26, 2018	DOWNLOAD
2018.1.7	181—182	Mar 21, 2018	DOWNLOAD
2017.3.15	173.1751—174	Mar 15, 2018	DOWNLOAD
2017.3.14	173.1751—174	Mar 15, 2018	DOWNLOAD
2018.1.6	181—182	Mar 13, 2018	DOWNLOAD
2018.1.4	181—182	Mar 05, 2018	DOWNLOAD
2018.1.3	181—182	Feb 20, 2018	DOWNLOAD