# Spark和Flume-ng整合

在本博客的《Spark读取Hbase中的数据》文章中我谈到了如何用Spark和Hbase整合的过程以及代码的编写测试等。今天我们继续谈谈Spark如何和Flume-ng进行整合，也就是如何将Flune-ng里面的数据发送到Spark，利用Spark进行实时的分析计算。本文将通过Java和Scala版本的程序进行程序的测试。

Spark和Flume-ng的整合属于Spark的Streaming这块。在讲述如何使用Spark Streaming之前，我们先来了解一下什么是Spark Streaming，在Spark官方文档是这么描述的（英文我就不翻译了，里面没有很复杂的语句）：

> Spark Streaming is an extension of the core Spark API that allows enables high-throughput, fault-tolerant stream processing of live data streams. Data can be ingested from many sources like Kafka, Flume, Twitter, ZeroMQ or plain old TCP sockets and be processed using complex algorithms expressed with high-level functions like map, reduce, joinand window. Finally, processed data can be pushed out to filesystems, databases, and live dashboards.

从上面的说明中我们可以看出Spark可以和Kafka、Flume、Twitter、ZeroMQ 和TCP sockets等进行整合，经过Spark处理，然后写入到filesystems、databases和live dashboards中。引用官方文档上面的一副图：



从上面的图片可以清楚的了解到各个模块所处的位置。这篇文章主要是讲述开发Spark Streaming这块，因为Flume-ng这块不需要特别的处理，完全和Flume-ng之间的交互一样。所有的Spark Streaming程序都是以JavaStreamingContext作为切入点的。如下：

```
JavaStreamingContext jssc =
    new JavaStreamingContext(master, appName,
                        new Duration(1000),
                        [sparkHome], [jars]);
JavaDStream<SparkFlumeEvent> flumeStream =
                        FlumeUtils.createStream(jssc, host, port);
```

最后需要调用JavaStreamingContext的start方法来启动这个程序。如下：

```
jssc.start();
jssc.awaitTermination();
```

整个程序如下：

```
package scala;

import org.apache.flume.source.avro.AvroFlumeEvent;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.function.Function;
import org.apache.spark.api.java.function.VoidFunction;
import org.apache.spark.storage.StorageLevel;
import org.apache.spark.streaming.Duration;
```

```
import org.apache.spark.streaming.api.java.JavaDStream;
import org.apache.spark.streaming.api.java.JavaStreamingContext;
import org.apache.spark.streaming.flume.FlumeUtils;
import org.apache.spark.streaming.flume.SparkFlumeEvent;

import java.nio.ByteBuffer;

/**
 * User: 过往记忆
 * Date: 14-7-8
 * Time: 下午23:16
 * bolg:
 * 本文地址：/archives/1063
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */
public static void JavaFlumeEventTest(String master, String host, int port) {
        Duration batchInterval = new Duration(2000);

        JavaStreamingContext ssc = new JavaStreamingContext(master,
                "FlumeEventCount", batchInterval,
                System.getenv("SPARK_HOME"),
                JavaStreamingContext.jarOfClass(JavaFlumeEventCount.class));
        StorageLevel storageLevel = StorageLevel.MEMORY_ONLY();
        JavaDStream<SparkFlumeEvent> flumeStream =
                FlumeUtils.createStream(ssc, host, port, storageLevel);

        flumeStream.count().map(new Function<java.lang.Long, String>() {
            @Override
            public String call(java.lang.Long in) {
                return "Received " + in + " flume events.";
            }
        }).print();

        ssc.start();
        ssc.awaitTermination();
}
```

然后开启Flume往这边发数据，在Spark的这端可以接收到数据：


如果你对Scala比较熟悉，下面是一段Scala的程序，功能和上面的一样：

```
import org.apache.spark.storage.StorageLevel
import org.apache.spark.streaming._
import org.apache.spark.streaming.flume._
import org.apache.spark.util.IntParam

/**
 * User: 过往记忆
 * Date: 14-7-8
 * Time: 下午23:16
 * bolg:
 * 本文地址：/archives/1063
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */

def ScalaFlumeEventTest(master :    String, host : String, port : Int) {
```

```
    val batchInterval = Milliseconds(2000)

    val ssc = new StreamingContext(master, "FlumeEventCount", batchInterval,
      System.getenv("SPARK_HOME"), StreamingContext.jarOfClass(this.getClass))

    val stream = FlumeUtils.createStream(ssc, host,port,StorageLevel.MEMORY_ONLY)

    stream.count().map(cnt => "Received " + cnt + " flume events." ).print()
    ssc.start()
    ssc.awaitTermination()
}
```

以上程序都是在Spark tandalone Mode下面运行的，如果你想在YARN上面运行，也是可以的，不过需要做点修改。具体怎么在Yarn上面运行，请参见官方文档。