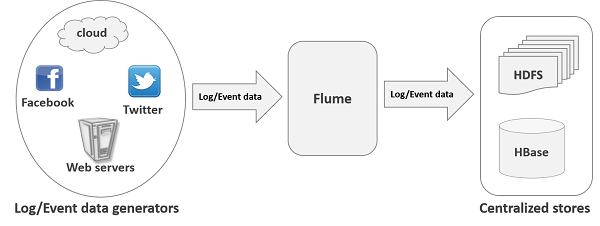
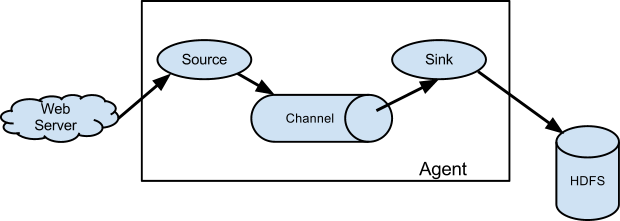
# Apache Flume的安装和使用

Apache Flume是可以收集日志、事件等数据资源，并将这些数据从各种数据源集中起来存储的工具/服务，其结构如下图：



其设计的原理是基于数据流，将日志数据从各种网站服务器上汇聚起来存储到HDFS/HBase等集中式存储器中。其系统架构如下：



其核心概念如下：

1. Events，数据传输的基本单位，其包括两个部分：header和body

* header，k/v结构数据，用来数据传输路由决策和其他结构化数据，如事件的时间戳或者数据源的服务器主机名，可以类比于HTTP Header的功能
* Body，字节数组，传输的实际数据

Flume为不同的Source生成的Evevent添加不同的Header

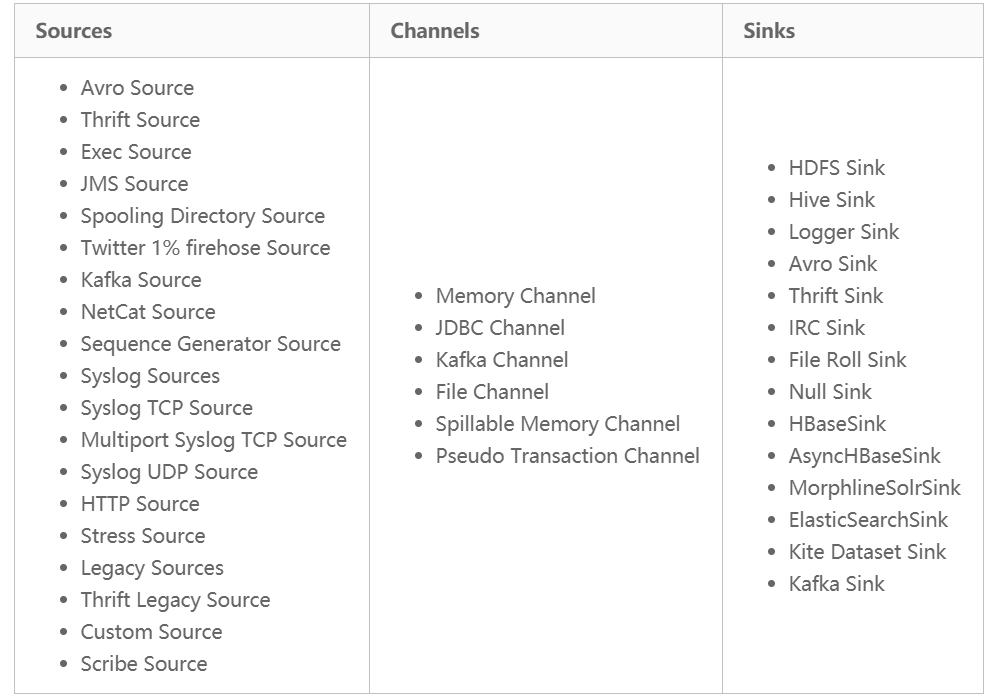
1. Agent, Flume代理，唯一启动的JVM进程，内部承载了从外部数据源事件流转到下一个目的地的过程，内部包含了Source/Channel/Sink
2. Source，从数据源收集数据，并传递给Channel，目前Source支持多种收集方式

* Avro，侦听Avro端口并从外部Avro客户端接收事件
* Spooling Directory Source，监控配置目录中新增文件，并从文件中读取数据
* NetCat Source，监听指定端口，将接收到的数据每一行转换为一个事件
* HTTP，接收HTTP GET/POST请求作为Flume事件
* Kafka Source，其作为Kafka消费者，从Kafka主题中读取消息
* 其他：Thrift,Exec,JMX,SysLog等

1. Channel，Event中转通道，临时存储Source组件传递过来的Event，其类似于消息队列，用于连接Source和Event
2. Sink，将Event从Channle取出后写入目标存储中，目前支持多种Sink

* HDFS Sink，写入到hadoop文件系统中，支持文件创建/序列文件等
* Hive Sink，将分割文件或者JSON数据的Events直接传送到Hive表或者分区中
* Logger Sink/File Roll Sink
* Avro /Thrift Sink/IRC Sink，写入服务端口
* HBase Sink，将数据写入到HBase中
* Kafka Sink，导出数据到Kafka Topic中
* 其他，ES/HTTP等，用户可以自定义Sink

Flume支持的Source、Sink及Channel类型如下：



可以根据需求搭配组合使用。

# 安装

1. 下载Flume

*wget http://apache.mirrors.hoobly.com/flume/1.8.0/apache-flume-1.8.0-bin.tar.gz*

1. 解压并修改配置

*tar -zxvf apache-flume-1.8.0-bin.tar.gz*

配置flume-env.sh，增加jdk配置

*export JAVA\_HOME=/usr/jdk64/jdk1.8.0\_112*

1. 验证是否安装成功

*bin/flume-ng version //输出如下*

*Flume 1.8.0*

*Source code repository: https://git-wip-us.apache.org/repos/asf/flume.git*

*Revision: 99f591994468633fc6f8701c5fc53e0214b6da4f*

*Compiled by denes on Fri Sep 15 14:58:00 CEST 2017*

*From source with checksum fbb44c8c8fb63a49be0a59e27316833d*

# 示例1：SpoolDir -> HDFS

示例如下：



1) 配置flume.conf，如下所示：

*// Agent中各组件名*

*a1.channels = ch1*

*a1.sinks = sink1*

*a1.sources = s1*

*// Source的配置，扫描文件并上传*

*a1.sources.s1.type = spooldir*

*a1.sources.s1.channels = ch1*

*a1.sources.s1.spoolDir = /opt/flume/data*

*//定义channel类型*

*a1.channels.ch1.type = memory*

*a1.channels.ch1.capacity = 1000*

*//描述，Sink，写入到hdfs的/tmp目录下，文件前缀test.txt*

*a1.sinks.sink1.type=hdfs*

*a1.sinks.sink1.hdfs.path=/tmp*

*a1.sinks.sink1.hdfs.filePrefix=test.txt*

*a1.sinks.sink1.channel=ch1*

1. 启动Flume Agent

*# bin/flume-ng agent -n a1 -c . -f conf/flume.conf -Dflume.root.logger=DEBUG,console*

启动日志如下：

*node.PollingPropertiesFileConfigurationProvider: Configuration provider starting*

*node.PollingPropertiesFileConfigurationProvider: Reloading configuration file:conf/flume-hdfs.conf*

*conf.FlumeConfiguration: Added sinks: sink1 Agent: a1*

*node.AbstractConfigurationProvider: Creating channels*

*channel.DefaultChannelFactory: Creating instance of channel ch1 type memory*

*node.AbstractConfigurationProvider: Created channel ch1*

*source.DefaultSourceFactory: Creating instance of source s1, type spooldir*

*sink.DefaultSinkFactory: Creating instance of sink: sink1, type: hdfs*

*node.Application: Starting Sink sink1*

*node.Application: Starting Source s1*

*hdfs.BucketWriter: Creating /tmp/test.txt.1542032999529.tmp*

1. 写入数据

向目录/opt/flume/data中写入数据

*echo “Hello Flume” >> /opt/flume/data/flume.data*

Agent输出日志如下：

*avro.ReliableSpoolingFileEventReader: Preparing to move file /opt/flume/data/flume.data to /opt/flume/data/flume.data2.COMPLETED*

1. 查看输出结果

*$ hdfs dfs -cat /tmp/test.txt.1542032999529*

*hello flume*

数据写入到hdfs中。

# 示例2，SpoolDir -> Kakfa



1. 配置Agent

*a1.channels = ch1*

*a1.sinks = sink1*

*a1.sources = s1*

*a1.sources.s1.type = spooldir*

*a1.sources.s1.channels = ch1*

*a1.sources.s1.spoolDir = /opt/flume/data*

*a1.channels.ch1.type = memory*

*a1.channels.ch1.capacity = 1000*

*a1.sinks.sink1.channel=ch1*

*a1.sinks.sink1.type=org.apache.flume.sink.kafka.KafkaSink*

*a1.sinks.sink1.kafka.topic=test*

*a1.sinks.sink1.kafka.bootstrap.servers=cmhhost1.novalocal:6667,cmhhost2.novalocal:6667,cmhhost3.novalocal:6667*

*a1.sinks.sink1.kafka.flumeBatchSize=1*

*a1.sinks.sink1.kafka.producer.acks=1*

1. 启动

*bin/flume-ng agent -n a1 -c . -f conf/flume-kafka.conf -Dflume.root.logger=DEBUG,console*

输出启动信息：

*8/11/13 16:51:30 INFO utils.AppInfoParser: Kafka version : 0.9.0.1*

*18/11/13 16:51:30 INFO utils.AppInfoParser: Kafka commitId : 23c69d62a0cabf06*

1. 测试

启动Kafka Consumer，topic会test，前期已经创建

*bin/kafka-console-consumer.sh --topic test --bootstrap-server cmhhost1.novalocal:6667,cmhhost2.novalocal:6667,cmhhost3.novalocal:6667*

写入数据

*echo "hello kafkatest">> fys.data*

1. 查看结果

*$ bin/kafka-console-consumer.sh --topic test --bootstrap-server cmhhost1.novalocal:6667,cmhhost2.novalocal:6667,cmhhost3.novalocal:6667*

*hello kafkatest*

# 示例3，自定义Sink

用户自定义Sink只需要继承基类AbstractSink，然后实现其中的方式即可，下面实现自定义Sink，将数据保存到定义文件

1. 添加依赖

*<dependencies>*

*<dependency>*

*<groupId>org.apache.flume</groupId>*

*<artifactId>flume-ng-core</artifactId>*

*<version>${flume.version}</version>*

*</dependency>*

*<dependency>*

*<groupId>org.apache.flume</groupId>*

*<artifactId>flume-ng-configuration</artifactId>*

*<version>${flume.version}</version>*

*</dependency>*

*</dependencies>*

1. 自定义Sink，继承AbstractSink，实现process方法

*public class MySink extends AbstractSink implements Configurable {*

*private static final Logger logger = LoggerFactory.getLogger(MySink.class);*

*private static final String PROP\_KEY\_ROOTPATH = "fileName";*

*private String fileName;*

*public void configure(Context context) {*

*fileName = context.getString(PROP\_KEY\_ROOTPATH);*

*}*

*public Status process() throws EventDeliveryException {*

*Channel ch = getChannel();*

*Transaction txn = ch.getTransaction();*

*txn.begin();*

*Event event = null;*

*while(true) {*

*event = ch.take();*

*if(event != null) {*

*break;*

*}}*

*try {*

*logger.info("Get Event");*

*String body = new String(event.getBody());*

*System.out.println("Event.getBody: "+ body);*

*String res = body + ":" + System.currentTimeMillis() + "\r\n";*

*File file = new File(fileName);*

*FileOutputStream fos = new FileOutputStream(file,true);*

*fos.write(res.getBytes());*

*fos.close();*

*txn.commit();*

*return Status.READY;*

*} catch (Exception ex) {*

*txn.rollback();*

*throw new EventDeliveryException(ex.getMessage());*

*} finally {*

*txn.close();*

*}*

*}*

*}*

1. 测试

* 打包成jar，放入lib中
* 配置文件

*a1.sources = s1*

*a1.channels = ch1*

*a1.sinks = sink1*

*a1.sources.s1.type = spooldir*

*a1.sources.s1.channels = ch1*

*a1.sources.s1.spoolDir=/opt/flume/data*

*a1.channels.ch1.type = memory*

*a1.channels.ch1.capacity = 1000*

*a1.sinks.sink1.type = com.fys.flume.sink.MySink*

*a1.sinks.sink1.fileName = /opt/flume/mysink/output.txt*

*a1.sinks.sink1.channel = ch1*

* 启动flume，进行测试，在/opt/flume/data中输入数据，查看/opt/flume/mysink/output.txt中添加数据如下：

*MySink:1542104841076*

# 其他

1. 定义Netcat Source

*a1.sources.s1.type = netcat*

*a1.sources.s1.bind = fys1.cmss.com*

*a1.sources.s1.port = 5678*

*a1.sources.s1.channels = ch1*

自定义Sink:

https://blog.csdn.net/harderxin/article/details/75032337

https://blog.csdn.net/liuxiao723846/article/details/78136079

https://www.cnblogs.com/netbloomy/p/6666683.html

http://flume.apache.org/releases/1.8.0.html