1.flume是什么？

flume是分布式的，可靠的，高可用的，用于对不同来源的大量的日志数据进行有效收集、聚集和移动，并以集中式的数据存储的系统。
整个系统分为三层：Agent层(客户机)，Collector层(中心服务器)和Store层(存储服务器)。

来自〈http://www.aboutyun.com/thread-8317-1-1.html〉

flume由agent组成,agent由source，channel，sink组成，Agent使用JVM 运行Flume

- source：source组件是专门用来收集数据的，可以处理各种类型、各种格式的日志数据,

(数据来源)包括avro、thrift、exec、jms、spooling directory、netcat、sequence generator、syslog、http、legacy、自定义

- channel：source组件把数据收集来以后，临时存放在channel中,即channel组件在agent中是专门用来存放临时数据的——对采集到的数据进行简单的缓存，可以存放在memory、jdbc、file等等。 它可以和任意数量的source和sink链接

> 来自 <http://www.cnblogs.com/gongxijun/p/5656778.html>

- sink：sink组件是用于把数据发送到目的地的组件，从channel接受内容,目的地包括hdfs、logger、avro、thrift、ipc、file、null、Hbase、solr、自定义。

> Thrift: thrift是一个软件框架，用来进行可扩展且跨语言的服务的开发(Facebook)。
> Avro:Avro是一个数据序列化系统，设计用于支持大批量数据交换的应用(Apache),flume内置

event：传输数据的基本单位。一个完整的event包括：event headers、event body、event信息(即文本文件中的单行记录)

event将传输的数据进行封装，如果是文本文件，通常是一行记录，

event也是事务的基本单位。event从source，流向channel，再到sink，

本身为一个字节数组并可携带headers(头信息)信息。

event代表着一个数据的最小完整单元，从外部数据源来，向外部的目的地去。

Flume提供了三种级别的可靠性保障，从强到弱依次分别为：

- **end-to-end**（收到数据agent首先将event写到磁盘上，当数据传送成功后，再删除；如果数据发送失败，可以重新发送。），
- **Store on failure**（这也是scribe采用的策略，当数据接收方crash时，将数据写到本地，待恢复后，继续发送），
- **Best effort**（数据发送到接收方后，不会进行确认）

来自〈http://www.cnblogs.com/oubo/archive/2012/05/25/2517751.html〉

2.关于flume启动

1)、flume组件启动顺序：channels——>sinks——>sources，关闭顺序：sources——>sinks——>channels；

2)、自动加载配置文件功能，会先关闭所有组件，再重启所有组件；

3)、关于AbstractConfigurationProvider中的Map<Class<? extends Channel>, Map<String, Channel>> channelCache这个对象，始终存储着agent中得所有channel对象，因为在动态加载时，channel中可能还有未消费完的数据，但是需要对channel重新配置，所以用以来缓存channel对象的所有数据及配置信息；

4)、通过在启动命令中添加"no-reload-conf"参数为true来取消自动加载配置文件功能；

来自〈http://www.cnblogs.com/lxf20061900/p/4012847.html〉

3. flume的运行过程

把数据从数据源(source)收集过来，在将收集到的数据送到指定的目的地(sink)。

为了保证输送的过程一定成功，在送到目的地(sink)之前，会先缓存数据(channel)，
待(全部)数据真正到达目的地(sink)后，flume在删除自己缓存的数据。

flume可以支持多级flume的agent，
即flume可以前后相继，例如sink可以将数据写到下一个agent的source中，这样的话就可以连成串了，可以整体处理了。flume还支持扇入(fan-in)、扇出(fan-out)。所谓扇入就是source可以接受多个输入，所谓扇出就是sink可以将数据输出多个目的地destination中。

来自〈http://blog.csdn.net/a2011480169/article/details/51544664〉