# Flume教程

flume是一个分布式、可靠、和高可用的海量日志采集、聚合和传输的系统。支持在日志系统中定制各类数据发送方，用于收集数据;同时，Flume提供对数据进行简单处理，并写到各种数据接收方(比如文本、HDFS、Hbase等)的能力。

flume的数据流由事件(Event)贯穿始终。事件是Flume的基本数据单位，它携带日志数据(字节数组形式)并且携带有头信息，这些Event由Agent外部的Source生成，当Source捕获事件后会进行特定的格式化，然后Source会把事件推入(单个或多个)Channel中。你可以把Channel看作是一个缓冲区，它将保存事件直到Sink处理完该事件。Sink负责持久化日志或者把事件推向另一个Source。

## 一、Flume特点

## 1、flume的可靠性：

当节点出现故障时，日志能够被传送到其他节点上而不会丢失。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），Besteffort（数据发送到接收方后，不会进行确认）。

## 2、flume的可恢复性：

还是靠Channel。推荐使用FileChannel，事件持久化在本地文件系统里(性能较差)。

## **3、flume的一些核心概念：**

Agent使用JVM 运行Flume。每台机器运行一个agent，但是可以在一个agent中包含多个sources和sinks。

Client生产数据，运行在一个独立的线程。

Source从Client收集数据，传递给Channel。

Sink从Channel收集数据，运行在一个独立线程。

Channel连接 sources 和 sinks ，这个有点像一个队列。

Events可以是日志记录、 avro 对象等。

Flume以agent为最小的独立运行单位。一个agent就是一个JVM。单agent由Source、Sink和Channel三大组件构成，如下图：

　　值得注意的是，Flume提供了大量内置的Source、Channel和Sink类型。不同类型的Source,Channel和Sink可以自由组合。组合方式基于用户设置的配置文件，非常灵活。比如：Channel可以把事件暂存在内存里，也可以持久化到本地硬盘上。Sink可以把日志写入HDFS, HBase，甚至是另外一个Source等等。Flume支持用户建立多级流，也就是说，多个agent可以协同工作，并且支持Fan-in、Fan-out、Contextual Routing、Backup Routes，这也正是NB之处。如下图所示:

二、如何安装？  
1.下载安装包

2.配置环境变量

3.修改配置文件(案例给出)

4.启动服务(案例给出)

5.验证   flume-ng -version

三、flume的案例  
案例1：Avro　可以发送一个给定的文件给Flume，Avro 源使用AVRO RPC机制

* 创建agent配置文件

|  |
| --- |
| vi avro.conf  a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= avro  a1.sources.r1.channels = c1  a1.sources.r1.bind = 0.0.0.0  a1.sources.r1.port = 4141  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

* 启动服务 flume agent a1

(d)使用avro-client发送文件

|  |  |
| --- | --- |
|  | flume-ng avro-client -c . -H m1 -p 4141 -F /home/hadoop/flume-1.5.0-bin/log.00 |

(f)在m1的控制台，可以看到以下信息，注意最后一行：  hello world

案例2：Spool 监测配置的目录下新增的文件，并将文件中的数据读取出来。需要注意两点：

1) 拷贝到spool目录下的文件不可以再打开编辑。  
2) spool目录下不可包含相应的子目录

(a)创建agent配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/spool.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= spooldir  a1.sources.r1.channels = c1  a1.sources.r1.spoolDir = /home/hadoop/flume-1.5.0-bin/logs  a1.sources.r1.fileHeader = true  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(b)启动服务flume agent a1

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/spool.conf -n a1 -Dflume.root.logger=INFO,console |

(c)追加文件到/home/hadoop/flume-1.5.0-bin/logs目录

|  |  |
| --- | --- |
|  | echo "spool test1" > /home/hadoop/flume-1.5.0-bin/logs/spool\_text.log |

(d)在m1的控制台，可以看到以下相关信息：

 Event: { headers:{file=/home/hadoop/flume-1.5.0-bin/logs/spool\_text.log} body: 73 70 6F 6F 6C 20 74 65 73 74 31        spool test1 }

案例3：Exec 执行一个给定的命令获得输出的源,如果要使用tail命令，必选使得file足够大才能看到输出内容

(a)创建agent配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/exec\_tail.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= exec  a1.sources.r1.channels = c1  a1.sources.r1.command= tail-F /home/hadoop/flume-1.5.0-bin/log\_exec\_tail  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(b)启动服务flume agent a1

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/exec\_tail.conf -n a1 -Dflume.root.logger=INFO,console |

(c)生成足够多的内容在文件里

|  |  |
| --- | --- |
|  | for i in {1..100};do echo "exec tail$i" >> /home/hadoop/flume-1.5.0-bin/log\_exec\_tail;echo $i;sleep 0.1;done |

(e)在m1的控制台，可以看到以下信息：

|  |  |
| --- | --- |
|  | Event: { headers:{} body: 65 78 65 63 20 74 61 69 6C 20 74 65 73 74 exec tail test }  Event: { headers:{} body: 65 78 65 63 20 74 61 69 6C 20 74 65 73 74 exec tail test } |

案例4：Syslogtcp 监听TCP的端口做为数据源  
(a)创建agent配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/syslog\_tcp.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= syslogtcp  a1.sources.r1.port = 5140  a1.sources.r1.host = localhost  a1.sources.r1.channels = c1  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |
|  |  |

　(b)启动flume agent a1

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/syslog\_tcp.conf -n a1 -Dflume.root.logger=INFO,console |

　(c)测试产生syslog

|  |  |
| --- | --- |
|  | echo "hello idoall.org syslog" | nc localhost 5140 |

　(d)在m1的控制台，可以看到以下信息：

|  |  |
| --- | --- |
|  | Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 68 65 6C 6C 6F 20 69 64 6F 61 6C 6C 2E 6F 72 67 hello idoall.org } |

案例5：JSONHandler

(a)创建agent配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/post\_json.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= org.apache.flume.source.http.HTTPSource  a1.sources.r1.port = 8888  a1.sources.r1.channels = c1  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(b)启动flume agent a1

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/post\_json.conf -n a1 -Dflume.root.logger=INFO,console |

(c)生成JSON 格式的POST request

|  |  |
| --- | --- |
|  | curl -X POST -d '[{ "headers" :{"a" : "a1","b" : "b1"},"body" : "idoall.org\_body"}]' [http://localhost:8888](http://localhost:8888/) |

(d)在m1的控制台，可以看到以下信息：

|  |  |
| --- | --- |
|  | Event: { headers:{b=b1, a=a1}  body: 69 64 6F 61 6C 6C 2E 6F 72 67 5F 62 6F 64 79  idoall.org\_body } |

案例6：Hadoop sink  
(a)创建agent配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/hdfs\_sink.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= syslogtcp  a1.sources.r1.port = 5140  a1.sources.r1.host = localhost  a1.sources.r1.channels = c1  # Describe the sink  a1.sinks.k1.type= hdfs  a1.sinks.k1.channel = c1  a1.sinks.k1.hdfs.path = hdfs://m1:9000/user/flume/syslogtcp  a1.sinks.k1.hdfs.filePrefix = Syslog  a1.sinks.k1.hdfs.round = true  a1.sinks.k1.hdfs.roundValue = 10  a1.sinks.k1.hdfs.roundUnit = minute  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(b)启动flume agent a1

|  |
| --- |
| flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/hdfs\_sink.conf -n a1 -Dflume.root.logger=INFO,console |

(c)测试产生syslog

|  |  |
| --- | --- |
|  | echo "hello idoall flume -> hadoop testing one" | nc localhost 5140 |

(d) 在m1上再打开一个窗口，去hadoop上检查文件是否生成

|  |  |
| --- | --- |
|  | hadoop fs -ls /user/flume/syslogtcp  hadoop fs -cat /user/flume/syslogtcp/Syslog.1407644509504 |

案例7：File Roll Sink  
(a)创建agent配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/file\_roll.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= syslogtcp  a1.sources.r1.port = 5555  a1.sources.r1.host = localhost  a1.sources.r1.channels = c1  # Describe the sink  a1.sinks.k1.type= file\_roll  a1.sinks.k1.sink.directory = /home/hadoop/flume-1.5.0-bin/logs  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(b)启动flume agent a1

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/file\_roll.conf -n a1 -Dflume.root.logger=INFO,console |

(c)测试产生log

|  |  |
| --- | --- |
|  | echo "hello idoall.org syslog" | nc localhost 5555  echo "hello idoall.org syslog 2" | nc localhost 5555 |

(d)查看/home/hadoop/flume-1.5.0-bin/logs下是否生成文件,默认每30秒生成一个新文件

|  |  |
| --- | --- |
|  | ll /home/hadoop/flume-1.5.0-bin/logs  cat /home/hadoop/flume-1.5.0-bin/logs/1407646164782-1  cat /home/hadoop/flume-1.5.0-bin/logs/1407646164782-2  hello idoall.org syslog  hello idoall.org syslog 2 |

案例8：Replicating Channel Selector Flume支持Fan out流从一个源到多个通道。有两种模式的Fan out，分别是复制和复用。在复制的情况下，流的事件被发送到所有的配置通道。在复用的情况下，事件被发送到可用的渠道中的一个子集。Fan out流需要指定源和Fan out通道的规则。这次我们需要用到m1,m2两台机器  
(a)在m1创建replicating\_Channel\_Selector配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector.conf    a1.sources = r1  a1.sinks = k1 k2  a1.channels = c1 c2  # Describe/configure the source  a1.sources.r1.type= syslogtcp  a1.sources.r1.port = 5140  a1.sources.r1.host = localhost  a1.sources.r1.channels = c1 c2  a1.sources.r1.selector.type= replicating  # Describe the sink  a1.sinks.k1.type= avro  a1.sinks.k1.channel = c1  a1.sinks.k1.hostname= m1  a1.sinks.k1.port = 5555  a1.sinks.k2.type= avro  a1.sinks.k2.channel = c2  a1.sinks.k2.hostname= m2  a1.sinks.k2.port = 5555  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  a1.channels.c2.type= memory  a1.channels.c2.capacity = 1000  a1.channels.c2.transactionCapacity = 100 |

(b)在m1创建replicating\_Channel\_Selector\_avro配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector\_avro.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= avro  a1.sources.r1.channels = c1  a1.sources.r1.bind = 0.0.0.0  a1.sources.r1.port = 5555  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(c)在m1上将2个配置文件复制到m2上一份

|  |  |
| --- | --- |
|  | scp -r /home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector.conf   scp -r /home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector\_avro.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector\_avro.conf |

(d)打开4个窗口，在m1和m2上同时启动两个flume agent

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector\_avro.conf -n a1 -Dflume.root.logger=INFO,console  flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/replicating\_Channel\_Selector.conf -n a1 -Dflume.root.logger=INFO,console |

(e)然后在m1或m2的任意一台机器上，测试产生syslog

|  |  |
| --- | --- |
|  | echo "hello idoall.org syslog" | nc localhost 5140 |

(f)在m1和m2的sink窗口，分别可以看到以下信息,这说明信息得到了同步：

|  |  |
| --- | --- |
|  | Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 68 65 6C 6C 6F 20 69 64 6F 61 6C 6C 2E 6F 72 67 hello idoall.org } |

案例9：Multiplexing Channel Selector  
(a)在m1创建Multiplexing\_Channel\_Selector配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector.conf    a1.sources = r1  a1.sinks = k1 k2  a1.channels = c1 c2  # Describe/configure the source  a1.sources.r1.type= org.apache.flume.source.http.HTTPSource  a1.sources.r1.port = 5140  a1.sources.r1.channels = c1 c2  a1.sources.r1.selector.type= multiplexing  a1.sources.r1.selector.header = type  #映射允许每个值通道可以重叠。默认值可以包含任意数量的通道。  a1.sources.r1.selector.mapping.baidu = c1  a1.sources.r1.selector.mapping.ali = c2  a1.sources.r1.selector.default = c1  # Describe the sink  a1.sinks.k1.type= avro  a1.sinks.k1.channel = c1  a1.sinks.k1.hostname= m1  a1.sinks.k1.port = 5555  a1.sinks.k2.type= avro  a1.sinks.k2.channel = c2  a1.sinks.k2.hostname= m2  a1.sinks.k2.port = 5555  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  a1.channels.c2.type= memory  a1.channels.c2.capacity = 1000  a1.channels.c2.transactionCapacity = 100 |

(b)在m1创建Multiplexing\_Channel\_Selector\_avro配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector\_avro.conf  a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= avro  a1.sources.r1.channels = c1  a1.sources.r1.bind = 0.0.0.0  a1.sources.r1.port = 5555  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(c)将2个配置文件复制到m2上一份

|  |  |
| --- | --- |
|  | scp -r /home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector.conf  scp -r /home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector\_avro.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector\_avro.conf |

(d)打开4个窗口，在m1和m2上同时启动两个flume agent

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector\_avro.conf -n a1 -Dflume.root.logger=INFO,console    flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/Multiplexing\_Channel\_Selector.conf -n a1 -Dflume.root.logger=INFO,console |

(e)然后在m1或m2的任意一台机器上，测试产生syslog

|  |  |
| --- | --- |
|  | curl -X POST -d '[{ "headers" :{"type" : "baidu"},"body" : "idoall\_TEST1"}]' [http://localhost:5140](http://localhost:5140/" \t "https://www.cnblogs.com/ciade/p/_blank) &&  curl -X POST -d '[{ "headers" :{"type" : "ali"},"body" : "idoall\_TEST2"}]' [http://localhost:5140](http://localhost:5140/" \t "https://www.cnblogs.com/ciade/p/_blank) &&  curl -X POST -d '[{ "headers" :{"type" : "qq"},"body" : "idoall\_TEST3"}]' [http://localhost:5140](http://localhost:5140/" \t "https://www.cnblogs.com/ciade/p/_blank) |

(f)在m1的sink窗口，可以看到以下信息：

|  |  |
| --- | --- |
|  | Event: { headers:{type=baidu} body: 69 64 6F 61 6C 6C 5F 54 45 53 54 31}  Event: { headers:{type=qq} body: 69 64 6F 61 6C 6C 5F 54 45 53 54 33} |

(g)在m2的sink窗口，可以看到以下信息：

|  |  |
| --- | --- |
|  | Event: { headers:{type=ali} body: 69 64 6F 61 6C 6C 5F 54 45 53 54 32} |

可以看到，根据header中不同的条件分布到不同的channel上

案例10：Flume Sink Processors failover的机器是一直发送给其中一个sink，当这个sink不可用的时候，自动发送到下一个sink。

(a)在m1创建Flume\_Sink\_Processors配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors.conf    a1.sources = r1  a1.sinks = k1 k2  a1.channels = c1 c2  #这个是配置failover的关键，需要有一个sink group  a1.sinkgroups = g1  a1.sinkgroups.g1.sinks = k1 k2  #处理的类型是failover  a1.sinkgroups.g1.processor.type= failover  #优先级，数字越大优先级越高，每个sink的优先级必须不相同  a1.sinkgroups.g1.processor.priority.k1 = 5  a1.sinkgroups.g1.processor.priority.k2 = 10  #设置为10秒，当然可以根据你的实际状况更改成更快或者很慢  a1.sinkgroups.g1.processor.maxpenalty = 10000  # Describe/configure the source  a1.sources.r1.type= syslogtcp  a1.sources.r1.port = 5140  a1.sources.r1.channels = c1 c2  a1.sources.r1.selector.type= replicating  # Describe the sink  a1.sinks.k1.type= avro  a1.sinks.k1.channel = c1  a1.sinks.k1.hostname= m1  a1.sinks.k1.port = 5555  a1.sinks.k2.type= avro  a1.sinks.k2.channel = c2  a1.sinks.k2.hostname= m2  a1.sinks.k2.port = 5555  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  a1.channels.c2.type= memory  a1.channels.c2.capacity = 1000  a1.channels.c2.transactionCapacity = 100 |

(b)在m1创建Flume\_Sink\_Processors\_avro配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors\_avro.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c  # Describe/configure the source  a1.sources.r1.type= avro  a1.sources.r1.channels = c1  a1.sources.r1.bind = 0.0.0.0  a1.sources.r1.port = 5555  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(c)将2个配置文件复制到m2上一份

|  |  |
| --- | --- |
|  | scp -r /home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors.conf    scp -r /home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors\_avro.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors\_avro.conf |

(d)打开4个窗口，在m1和m2上同时启动两个flume agent

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors\_avro.conf -n a1 -Dflume.root.logger=INFO,console  flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors.conf -n a1 -Dflume.root.logger=INFO,console |

(e)然后在m1或m2的任意一台机器上，测试产生log

|  |  |
| --- | --- |
|  | echo "idoall.org test1 failover" | nc localhost 5140 |

(f)因为m2的优先级高，所以在m2的sink窗口，可以看到以下信息，而m1没有：

|  |  |
| --- | --- |
|  | Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 31 idoall.org test1 } |

(g)这时我们停止掉m2机器上的sink(ctrl+c)，再次输出测试数据：

|  |  |
| --- | --- |
|  | echo "idoall.org test2 failover" | nc localhost 5140 |

(h)可以在m1的sink窗口，看到读取到了刚才发送的两条测试数据：

|  |  |
| --- | --- |
|  | Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 31 idoall.org test1 }  Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 32 idoall.org test2 } |

(i)我们再在m2的sink窗口中，启动sink：

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/Flume\_Sink\_Processors\_avro.conf -n a1 -Dflume.root.logger=INFO,console |

(j)输入两批测试数据：

|  |  |
| --- | --- |
|  | echo "idoall.org test3 failover" | nc localhost 5140 && echo "idoall.org test4 failover" | nc localhost 5140 |

(k)在m2的sink窗口,我们可以看到以下信息,因为优先级的关系,log消息会再次落到m2上：

|  |  |
| --- | --- |
|  | Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 33 idoall.org test3 }  Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 34 idoall.org test4 } |

案例11：Load balancing Sink Processor load balance type和failover不同的地方是,load balance有两个配置,一个是轮询,一个是随机。两种情况下如果被选择的sink不可用,就会自动尝试发送到下一个可用的sink上面。

(a)在m1创建Load\_balancing\_Sink\_Processors配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors.conf    a1.sources = r1  a1.sinks = k1 k2  a1.channels = c1  #这个是配置Load balancing的关键，需要有一个sink group  a1.sinkgroups = g1  a1.sinkgroups.g1.sinks = k1 k2  a1.sinkgroups.g1.processor.type= load\_balance  a1.sinkgroups.g1.processor.backoff = true  a1.sinkgroups.g1.processor.selector = round\_robin  # Describe/configure the source  a1.sources.r1.type= syslogtcp  a1.sources.r1.port = 5140  a1.sources.r1.channels = c1  # Describe the sink  a1.sinks.k1.type= avro  a1.sinks.k1.channel = c1  a1.sinks.k1.hostname= m1  a1.sinks.k1.port = 5555  a1.sinks.k2.type= avro  a1.sinks.k2.channel = c1  a1.sinks.k2.hostname= m2  a1.sinks.k2.port = 5555  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100 |

(b)在m1创建Load\_balancing\_Sink\_Processors\_avro配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors\_avro.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= avro  a1.sources.r1.channels = c1  a1.sources.r1.bind = 0.0.0.0  a1.sources.r1.port = 5555  # Describe the sink  a1.sinks.k1.type= logger  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(c)将2个配置文件复制到m2上一份

|  |  |
| --- | --- |
|  | scp -r /home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors.conf    scp -r /home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors\_avro.conf root@m2:/home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors\_avro.conf |

(d)打开4个窗口，在m1和m2上同时启动两个flume agent

|  |  |
| --- | --- |
|  | flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors\_avro.conf -n a1 -Dflume.root.logger=INFO,console  flume-ng agent -c . -f /home/hadoop/flume-1.5.0-bin/conf/Load\_balancing\_Sink\_Processors.conf -n a1 -Dflume.root.logger=INFO,console |

(e)然后在m1或m2的任意一台机器上，测试产生log，一行一行输入，输入太快，容易落到一台机器上

|  |  |
| --- | --- |
|  | echo "idoall.org test1" | nc localhost 5140  echo "idoall.org test2" | nc localhost 5140  echo "idoall.org test3" | nc localhost 5140  echo "idoall.org test4" | nc localhost 5140 |

(f)在m1的sink窗口，可以看到以下信息：

|  |  |
| --- | --- |
|  | Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 32 idoall.org test2 }  Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 34 idoall.org test4 } |

(g)在m2的sink窗口，可以看到以下信息：

|  |  |
| --- | --- |
|  | Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 31 idoall.org test1 }  Event: { headers:{Severity=0, flume.syslog.status=Invalid, Facility=0} body: 69 64 6F 61 6C 6C 2E 6F 72 67 20 74 65 73 74 33 idoall.org test3 } |

　说明轮询模式起到了作用。

案例12：Hbase sink  
 (a)在测试之前，请先将hbase启动

(b)然后将以下文件复制到flume中：

|  |  |
| --- | --- |
|  | cp/home/hadoop/hbase-0.96.2-hadoop2/lib/protobuf-java-2.5.0.jar /home/hadoop/flume-1.5.0-bin/lib  cp/home/hadoop/hbase-0.96.2-hadoop2/lib/hbase-client-0.96.2-hadoop2.jar /home/hadoop/flume-1.5.0-bin/lib  cp/home/hadoop/hbase-0.96.2-hadoop2/lib/hbase-common-0.96.2-hadoop2.jar /home/hadoop/flume-1.5.0-bin/lib  cp/home/hadoop/hbase-0.96.2-hadoop2/lib/hbase-protocol-0.96.2-hadoop2.jar /home/hadoop/flume-1.5.0-bin/lib  cp/home/hadoop/hbase-0.96.2-hadoop2/lib/hbase-server-0.96.2-hadoop2.jar /home/hadoop/flume-1.5.0-bin/lib  cp/home/hadoop/hbase-0.96.2-hadoop2/lib/hbase-hadoop2-compat-0.96.2-hadoop2.jar /home/hadoop/flume-1.5.0-bin/lib  cp/home/hadoop/hbase-0.96.2-hadoop2/lib/hbase-hadoop-compat-0.96.2-hadoop2.jar /home/hadoop/flume-1.5.0-bin/lib  cp/home/hadoop/hbase-0.96.2-hadoop2/lib/htrace-core-2.04.jar /home/hadoop/flume-1.5.0-bin/lib |

(c)确保test\_idoall\_org表在hbase中已经存在。

(d)在m1创建hbase\_simple配置文件

|  |  |
| --- | --- |
|  | vi /home/hadoop/flume-1.5.0-bin/conf/hbase\_simple.conf    a1.sources = r1  a1.sinks = k1  a1.channels = c1  # Describe/configure the source  a1.sources.r1.type= syslogtcp  a1.sources.r1.port = 5140  a1.sources.r1.host = localhost  a1.sources.r1.channels = c1  # Describe the sink  a1.sinks.k1.type= logger  a1.sinks.k1.type= hbase  a1.sinks.k1.table = test\_idoall\_org  a1.sinks.k1.columnFamily = name  a1.sinks.k1.column = idoall  a1.sinks.k1.serializer = org.apache.flume.sink.hbase.RegexHbaseEventSerializer  a1.sinks.k1.channel = memoryChannel  # Use a channel which buffers events in memory  a1.channels.c1.type= memory  a1.channels.c1.capacity = 1000  a1.channels.c1.transactionCapacity = 100  # Bind the source and sink to the channel  a1.sources.r1.channels = c1  a1.sinks.k1.channel = c1 |

(e)启动flume agent

|  |  |
| --- | --- |
|  | flume-ngagent -c . –f /home/hadoop/flume-1.5.0-bin/conf/hbase\_simple.conf -n a1 -Dflume.root.logger=INFO,console |

(f)测试产生syslog

|  |  |
| --- | --- |
|  | echo "hello idoall.org from flume" | nc localhost 5140 |

(g)这时登录到hbase中，可以发现新数据已经插入

|  |
| --- |
| hbase shell    hbase(main):001:0> list  TABLE  hbase2hive\_idoall  hive2hbase\_idoall  test\_idoall\_org    => ["hbase2hive\_idoall","hive2hbase\_idoall","test\_idoall\_org"]    hbase(main):002:0> scan "test\_idoall\_org"    hbase(main):004:0> quit |