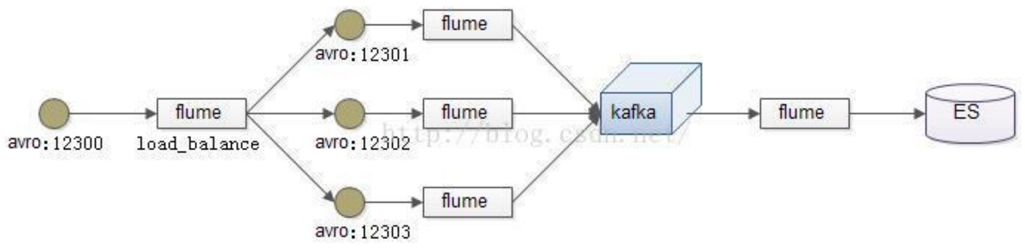
flume 的负载均衡 load balancer

负载均衡是用于解决一台机器(一个进程)无法解决所有请求而产生的一种算法。Load balancing Sink Processor 能够实现 load balance 功能，如下图 Agent1 是一个路由节点，负责将 Channel 暂存的 Event 均衡到对应的多个 Sink 组件上，而每个 Sink 组件分别连接到一个独立的 Agent 上，示例配置，如下所示：



在此处我们通过三台机器来进行模拟 flume 的负载均衡
三台机器规划如下：
node01：采集数据，发送到 node02 和 node03 机器上去
node02：接收 node01 的部分数据
node03：接收 node01 的部分数据

第一步：开发 node01 服务器的 flume 配置

node01 服务器配置：

```
cd /export/servers/apache-flume-1.6.0-cdh5.14.0-bin/conf
vim load_banlancer_client.conf
```

```
#agent name
a1.channels = c1
a1.sources = r1
a1.sinks = k1 k2

#set gruop
a1.sinkgroups = g1

#set channel
a1.channels.c1.type = memory
```

```
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sources.r1.channels = c1
a1.sources.r1.type = exec
a1.sources.r1.command = tail -F /export/servers/taillogs/access_log

# set sink1
a1.sinks.k1.channel = c1
a1.sinks.k1.type = avro
a1.sinks.k1.hostname = node02
a1.sinks.k1.port = 52020

# set sink2
a1.sinks.k2.channel = c1
a1.sinks.k2.type = avro
a1.sinks.k2.hostname = node03
a1.sinks.k2.port = 52020

#set sink group
a1.sinkgroups.g1.sinks = k1 k2

#set failover
a1.sinkgroups.g1.processor.type = load_balance
a1.sinkgroups.g1.processor.backoff = true
a1.sinkgroups.g1.processor.selector = round_robin
a1.sinkgroups.g1.processor.selector.maxTimeOut=10000
```

第二步：开发 node02 服务器的 flume 配置

```
cd /export/servers/apache-flume-1.6.0-cdh5.14.0-bin/conf
vim load_banlancer_server.conf
```

```
# Name the components on this agent
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 = node02
a1.sources.r1.port = 52020

# 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
```

第三步：开发 node03 服务器 flume 配置

node03 服务器配置

```
cd /export/servers/apache-flume-1.6.0-cdh5.14.0-bin/conf
vim load_banlancer_server.conf
```

```
# Name the components on this agent
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 = node03
a1.sources.r1.port = 52020

# 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
```

第四步：准备启动 flume 服务

启动 node03 的 flume 服务

```
cd /export/servers/apache-flume-1.6.0-cdh5.14.0-bin
bin/flume-ng agent -n a1 -c conf -f conf/load_banlancer_server.conf -
Dflume.root.logger=DEBUG,console
```

启动 node02 的 flume 服务

```
cd /export/servers/apache-flume-1.6.0-cdh5.14.0-bin
bin/flume-ng agent -n a1 -c conf -f conf/load_banlancer_server.conf -
Dflume.root.logger=DEBUG,console
```

启动 node01 的 flume 服务

```
cd /export/servers/apache-flume-1.6.0-cdh5.14.0-bin
bin/flume-ng agent -n a1 -c conf -f conf/load_banlancer_client.conf -
Dflume.root.logger=DEBUG,console
```

第五步：node01 服务器运行脚本产生数据

```
cd /export/servers/shells
sh tail-file.sh
```