Channel 处理器调用 `setChannels` 方法，该方法传递给所有的 Channel，即选择器必须为每个事件选择的 Channel。该类实现了 `Configurable` 接口，所以当选择器初始化时调用 `configure` 方法。当每个事件将要处理时，调用 `getRequiredChannels` 和 `getOptionalChannels` 方法。`getAllChannels` 方法必须返回在创建期间 Channel 处理器设置的所有 Channel。

这个类也提供了一对便捷的方法：一个方法返回 Channel 名字的一个 map，给实际的 Channel 实例；另一个方法返回 Channel 实例的列表，该列表是以空格分隔的字符串表示的 Channel 名字。可以使用 FQCN 部署 Channel 选择器：

157

```
agent.sources.avroSrc.type = avro
agent.sources.avroSrc.channels = c1 c2 c3 c4 c5
agent.sources.avroSrc.selector.type = com.usingflume.selector.RandomSelector
agent.sources.avroSrc.selector.default = c5
agent.sources.avroSrc.selector.random.seed = 4532
```

在这种情况下，自定义选择器获取通过 `agent.sources.avro.selector.` 传递的所有参数，就像其他组件一样。在这个例子中，选择器将会在键为 `default` 和 `random.seed`，值分别为 c5 和 4532 的 configure 方法中，得到一个 Context 实例。

自定义 Channel 选择器应该放入 plugins.d 目录，就像第 8 章“部署自定义代码”一节描述的那样。

Sink 组和 Sink 处理器

第 5 章我们已经讨论了 Sink 是如何工作的，以及 Flume 自带的不同种类的 Sink 处理器。我们也简要讨论过 Sink 组和 Sink 处理器。正如我们前面讨论的，Flume 配置框架为每个 Sink 组实例化一个 Sink 运行器，来运行 Sink 组。每个 Sink 组可以包含任意数量的 Sink。Sink 运行器持续请求 Sink 组，要求其中的一个 Sink 从自己的 Channel 中读取事件。Sink 组通常用于 RPC Sink，在层之间以负载均衡或故障转移方式发送数据。

因为 RPC Sink 被设计成只连接一个 RPC Source，从一个 Flume Agent 发送数据到下一层的一组 Agent，至少需要和发送事件到 Agent 一样多的 Sink。为了确保每个 Agent 发送事件到下一层的多个目的地 Agent，且一层上的每层发送数据到下一层的所有 Agent，不需要压倒性地占据网络或那些 Agent，那么每个 Agent 可以在下一层所有机器之间负载均衡。