续表

配置参数	默认值	描述
selector.maxTimeOut	30000	毫秒值时间，该时间之后黑名单时间周期不再增长

load-balancing sink 处理器用下面的方式进行配置：

```
agent.sinks = s1 s2 s3 s4
agent.sinkgroups = sg1
agent.sinkgroups.sg1.sinks = s1 s2 s3 s4
agent.sinkgroups.sg1.processor.type = load_balance
agent.sinkgroups.sg1.processor.selector = random
agent.sinkgroups.sg1.processor.backoff = true
agent.sinkgroups.sg1.processor.selector.maxTimeOut = 10000
```


该配置设置 Sink 组使用 load-balancing sink 处理器，随机选择 s1、s2、s3 或 s4 中的一个。如果一个 Sink（或者更准确地，Sink 发送数据到的 Agent）失败，那么该 Sink 会被加入到黑名单，回退时间从 250 毫秒开始，然后以指数形式增长直到达到 10 秒。在这之后，每次写操作失败，Sink 就回退 10 秒，直到它能够成功写入数据，此时回退时间被重置为 0。如果 selector 参数值设置为 round_robin，那么 s1 被首先用来处理数据，然后是 s2，然后是 s3，接下来是 s4，然后再次是 s1。

该配置意味着，在任何时候每个 Agent 只有一个 Sink 写数据。可以通过添加多个有相似配置的 load-balancing sink 处理器的 Sink 组进行修改。注意，可能会有多个 Agent 尝试写入数据到第二层每个 Agent。

拥有太多 Sink 发送数据到相同 Agent 的风险

因为每个 Avro Sink 对 Avro Source 保持持续开放的连接，拥有写入到相同 Agent 的多个 Sink 会增加更多的 socket 连接，且在第二层 Agent 上占据更多的资源。对相同 Agent 增加大量 Sink 之前必须要谨慎考虑。

编写 Sink 选择器 *

每次 Sink 运行器调用 process 方法时，可以使用自定义逻辑的 load-balancing sink 处理器，用来选择激活使用哪个 Sink。自定义选择器必须实现这里展示的 LoadBalancingSinkProcessor$SinkSelector 接口：

```
public interface SinkSelector extends Configurable, LifecycleAware {
  void setSinks(List<Sink> sinks);
  Iterator<Sink> createSinkIterator();
```