例如，一个优先级为 100 的 Sink 在优先级为 90 的 Sink 之前被激活。如果没有为指定的 Sink 设置优先级，那么 Sink 的优先级取决于 Sink 在 Sink 组配置中被指定的顺序。每次 Sink 写数据失败时，Sink 被认为失败且在短暂的一段时间内被加入黑名单。这个黑名单中的时间间隔（与 load-balancing sink 处理器的回退时间类似）在每个连续导致失败的 164 尝试时增加，直到达到 maxpenalty 指定的值（以毫秒为单位）。一旦黑名单时间间隔达到该值，在尝试了这么多毫秒之后，将导致 Sink 进一步的失败。一旦 Sink 在这么多时间之后成功写入数据，那么回退时间被重置为 0。看一下下面的配置：

```
agent.sinks = s1 s2 s3 s4
agent.sinkgroups.sg1.sinks = s1 s2 s3 s4
agent.sinkgroups.sg1.processor.type = failover
agent.sinkgroups.sg1.processor.priority.s2 = 100
agent.sinkgroups.sg1.processor.priority.s1 = 90
agent.sinkgroups.sg1.processor.priority.s4 = 110
agent.sinkgroups.sg1.processor.maxpenalty = 10000
```

在该配置中，四个 Sink 使用了 failover 的配置，Sink s4 有最高的优先级，接下来是 s2 和 s1。Sink s3 没有设置优先级。对于没有指定优先级的 Sink，第一个没有优先级的 Sink 被分配优先级 0，下一个分配优先级 -1，下一个分配 -2，以此类推。这些优先级只分配给没有设置优先级的 Sink。因此，这里展示的配置样本间接分配了 s3 优先级 0，所以 Sink 以 s4、s2、s1、s3 的顺序进行重试。值得注意的是，如果两个 Sink 有相同的优先级（间接或明确分配的），只激活 Sink 组中首先指定的 Sink。也要注意，如果明确和间接的优先级被设置为相同的范围，那么它们的值也和上面一样使用。例如：

```
agent.sinks = s1 s2 s3 s4
agent.sinkgroups.sg1.sinks = s1 s2 s3 s4 s5 s6
agent.sinkgroups.sg1.processor.type = failover
agent.sinkgroups.sg1.processor.priority.s2 = 0
agent.sinkgroups.sg1.processor.priority.s4 = 110
agent.sinkgroups.sg1.processor.priority.s5 = -5
agent.sinkgroups.sg1.processor.priority.s6 = -2
agent.sinkgroups.sg1.processor.maxpenalty = 10000
```

在该配置中，Sink s4 有最高的优先级，所以 s4 最先被激活。Sink s1 将被分配优先级 0——和 s2 相同——这意味着 s2 不会被激活。s3 的优先级是 -1，所以激活的顺序是 s4、s1、s3、s6、s5。虽然 s3 的优先级在配置中没有指定，但是间接指定了它的优先级比 s5 和 s6 的高，所以 s3 在它们两个之前被激活。