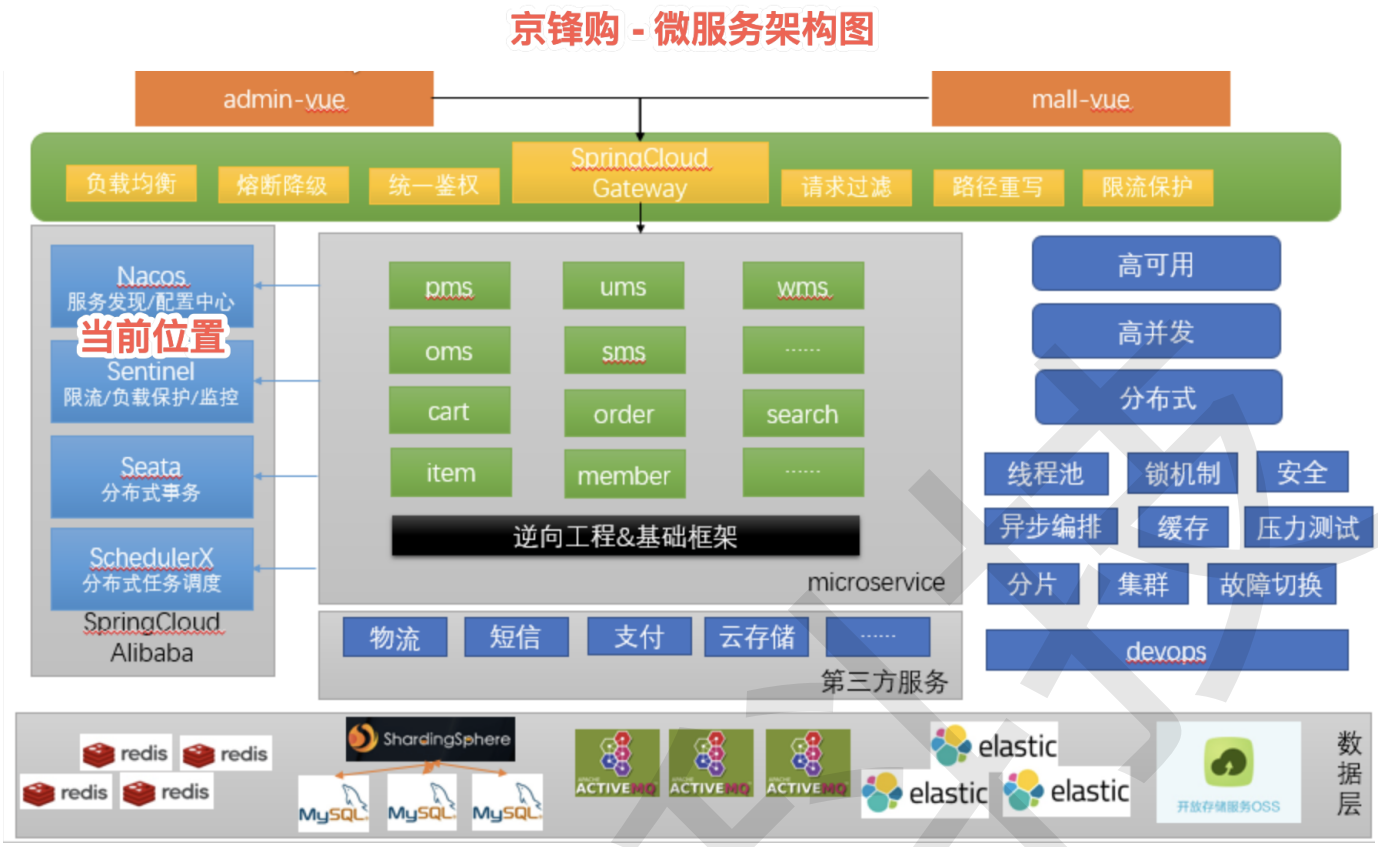
微服务项目-京锋购 04 - 流量控制与熔断降级

京锋购

京东 X 砺锋 = 京锋购商城



流量控制

什么是流量控制

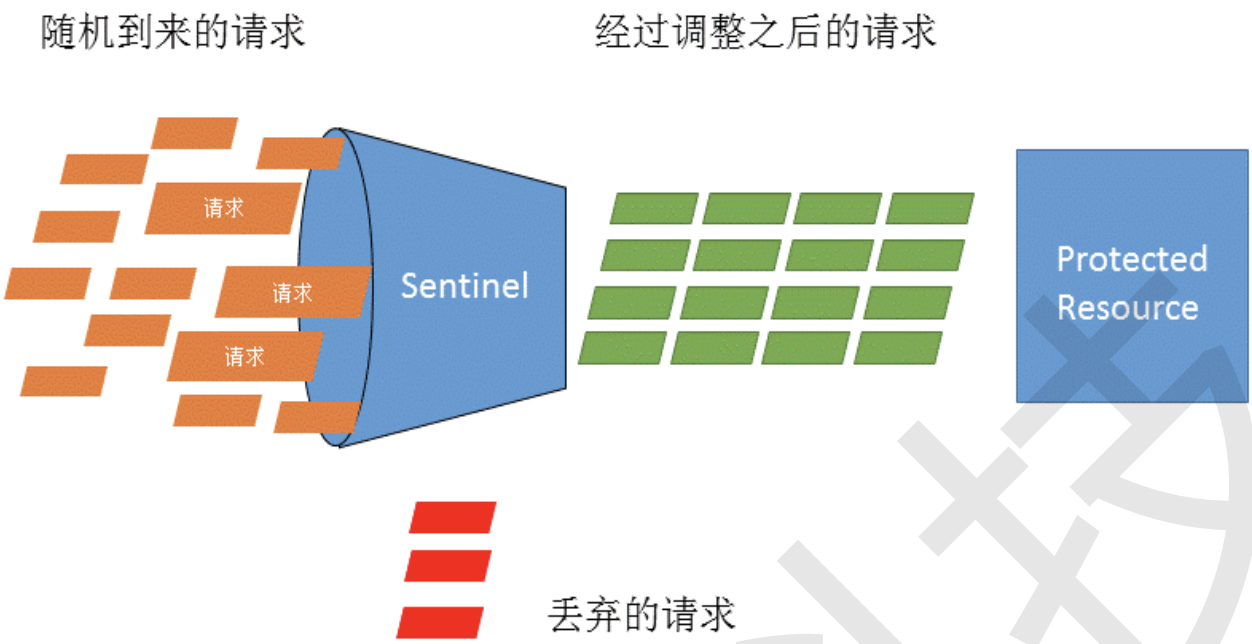
流量控制在网络传输中是一个常用的概念，它用于调整网络包的发送数据。

然而，从系统稳定性角度考虑，在处理请求的速度上，也有非常多的讲究。

任意时间到来的请求往往是随机不可控的，而系统的处理能力是有限的。

我们需要根据系统的处理能力对流量进行控制。

Sentinel 作为一个调配器，可以根据需要把随机的请求调整成合适的形状，如下图所示：



流量控制设计理念，流量控制有以下几个角度:

- 资源的调用关系，例如资源的调用链路，资源和资源之间的关系；
- 运行指标，例如 QPS、线程池、系统负载等；
- 控制的效果，例如直接限流、冷启动、排队等。

Sentinel 的设计理念是让您自由选择控制的角度，并进行灵活组合，从而达到想要的效果。

比如对于流量就有如下配置：

QPS流量控制

当 QPS 超过某个阈值的时候，则采取措施进行流量控制。

流控模式有以下几种：

- 直接
- 关联
- 链路

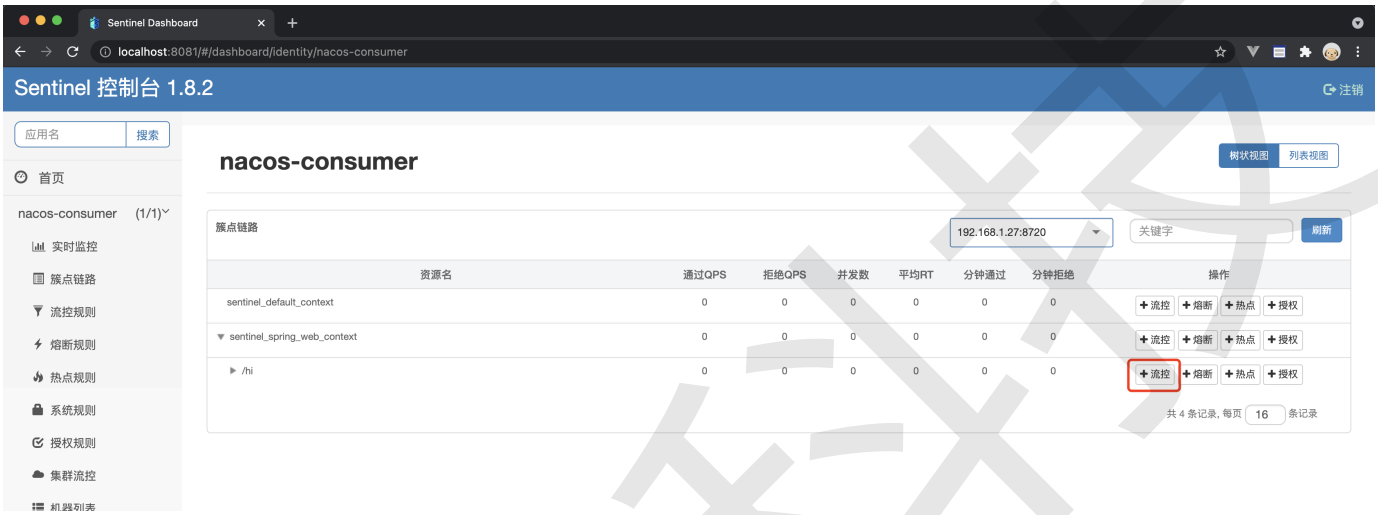
流控效果包括以下几种：

- 直接拒绝
- Warm Up
- 匀速排队

直接拒绝

直接拒绝（RuleConstant.CONTROL_BEHAVIOR_DEFAULT）方式是默认的流量控制方式，当QPS超过任意规则的阈值后，新的请求就会被立即拒绝，拒绝方式为抛出 FlowException。

这种方式适用于对系统处理能力确切已知的情况下，比如通过压测确定了系统的准确水位。



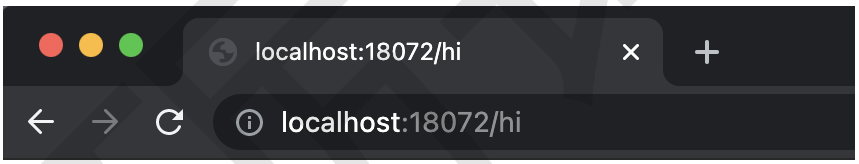
现在做一个最简单的配置：

- 阈值类型选择：QPS
- 单机阈值：2

综合起来的配置效果就是，该接口的限流策略是每秒最多允许2个请求进入。

点击新增按钮之后，可以看到如下界面：

在浏览器访问：http://localhost:18072/hi，疯狂刷新，将出现如下信息：




```
Blocked by Sentinel (flow limiting)
```

Warm Up（预热）

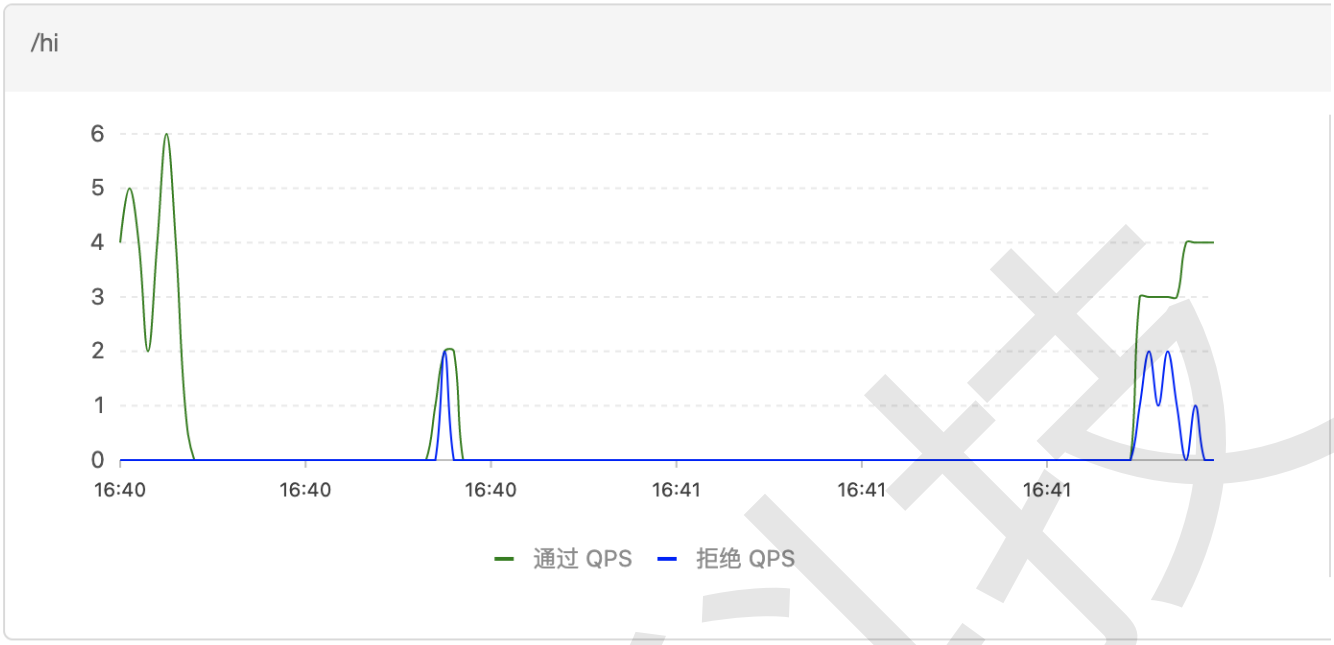
Warm Up（`RuleConstant.CONTROL_BEHAVIOR_WARM_UP`）方式，即预热/冷启动方式。

当系统长期处于低水位的情况下，当流量突然增加时，直接把系统拉升到高水位可能瞬间把系统压垮。

通过"冷启动"，让通过的流量缓慢增加，在一定时间内逐渐增加到阈值上限，给冷系统一个预热的时间，避免冷系统被压垮。

此时如果疯狂刷新，可以发现前几秒会发生熔断，几秒钟之后就完全没有问题了

实时监控



匀速排队

匀速排队（`RuleConstant.CONTROL_BEHAVIOR_RATE_LIMITER`）方式会严格控制请求通过的间隔时间，也即是让请求以均匀的速度通过，对应的是漏桶算法。

测试配置如下：1s 处理一个请求，排队等待，等待时间 10s 。

关联限流模式

关联限流：当关联的资源请求达到阈值时，就限流自己。

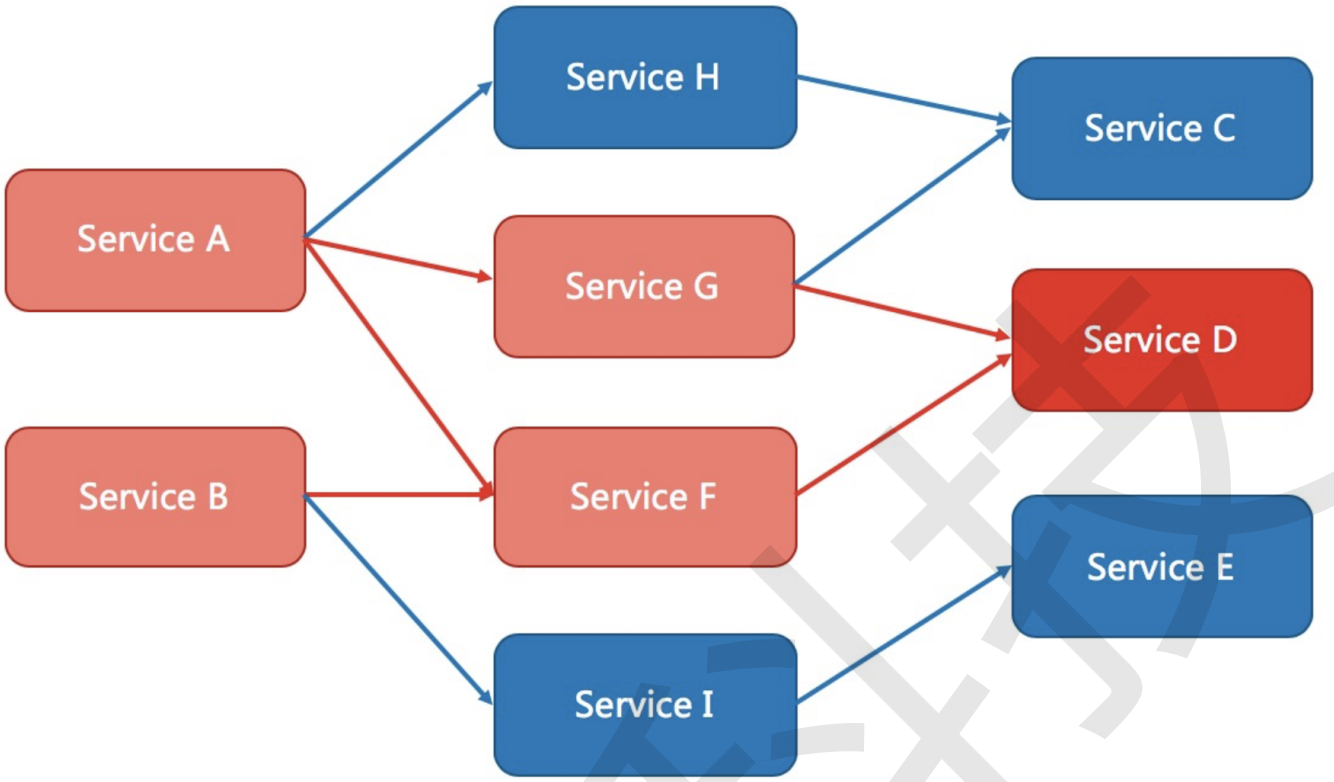
比如配置如下：/hi2 的关联资源 /hi，并发数超过 2 时，/hi2 就限流自己

熔断降级

什么是熔断降级

除了流量控制以外，及时对调用链路中的不稳定因素进行熔断也是 Sentinel 的使命之一，也是保障高可用的重要措施之一。

由于调用关系的复杂性，如果调用链路中的某个资源出现了不稳定，可能会导致请求发生堆积，进而导致级联错误。



Sentinel 和 Hystrix 的原则是一致的: 当检测到调用链路中某个资源出现不稳定的表现，例如请求响应时间长或异常比例升高的时候，则对这个资源的调用进行限制，让请求快速失败，避免影响到其它的资源而导致级联故障。

熔断降级设计理念

在限制的手段上，Sentinel 和 Hystrix 采取了完全不一样的方法。

Hystrix 通过 线程池隔离 的方式，来对依赖（在 Sentinel 的概念中对应 资源）进行了隔离。这样做的好处是资源和资源之间做到了最彻底的隔离。缺点是除了增加了线程切换的成本（过多的线程池导致线程数目过多），还需要预先给各个资源做线程池大小的分配。

Sentinel 对这个问题采取了两种手段:

- 通过并发线程数进行限制

和资源池隔离的方法不同，Sentinel 通过限制资源并发线程的数量，来减少不稳定资源对其它资源的影响。
这样不但没有线程切换的损耗，也不需要您预先分配线程池的大小。

当某个资源出现不稳定的情况下，例如响应时间变长，对资源的直接影响就是会造成线程数的逐步堆积。
当线程数在特定资源上堆积到一定的数量之后，对该资源的新请求就会被拒绝。
堆积的线程完成任务后才开始继续接收请求。

- 通过响应时间对资源进行降级

除了对并发线程数进行控制以外，Sentinel 还可以通过响应时间来快速降级不稳定的资源。
当依赖的资源出现响应时间过长后，所有对该资源的访问都会被直接拒绝，直到过了指定的时间窗口之后才重新恢复。

限流降级指标有三个：

1. 慢调用比例（平均响应时间 – RT）
2. 异常比例
3. 异常数

慢调用比例 (SLOW_REQUEST_RATIO)

选择以慢调用比例作为阈值，需要设置允许的慢调用 RT（即最大的响应时间），请求的响应时间大于该值则统计为慢调用。当单位统计时长（statIntervalMs）内请求数目大于设置的最小请求数目，并且慢调用的比例大于阈值，则接下来的熔断时长内请求会自动被熔断。

经过熔断时长后熔断器会进入探测恢复状态（HALF-OPEN 状态），若接下来的一个请求响应时间小于设置的慢调用 RT 则结束熔断，若大于设置的慢调用 RT 则会再次被熔断。

比如，当资源的平均响应时间超过阈值（DegradeRule 中的 count，以 ms 为单位，默认上限是 4900ms）之后，资源进入准降级状态。

如果1s之内持续进入 5 个请求，它们的 RT 都持续超过这个阈值，那么在接下来的时间窗口（DegradeRule 中的 timeWindow，以 s 为单位）之内，对这个方法的调用都会自动地返回（抛出 DegradeException）。

在下一个时间窗口到来时, 会接着再放入5个请求, 再重复上面的判断。

异常比例 (ERROR_RATIO)

当单位统计时长（statIntervalMs）内请求数目大于设置的最小请求数目，并且异常的比例大于阈值，则接下来的熔断时长内请求会自动被熔断。

经过熔断时长后熔断器会进入探测恢复状态（HALF-OPEN 状态），若接下来的一个请求成功完成（没有错误）则结束熔断，否则会再次被熔断。异常比率的阈值范围是 [0.0, 1.0]，代表 0% - 100%。

比如，当资源的每秒请求量 >= 5，且每秒异常总数占通过量的比值超过阈值（DegradeRule 中的 count）之后，资源进入降级状态，即在接下的时间窗口（DegradeRule中的 timeWindow，以 s 为单位）之内，对这个方法的调用都会自动地返回。

异常比率的阈值范围是 [0.0, 1.0]，代表 0% - 100%。

异常数 (ERROR_COUNT)

当单位统计时长内的异常数目超过阈值之后会自动进行熔断。

经过熔断时长后熔断器会进入探测恢复状态（HALF-OPEN 状态），若接下来的一个请求成功完成（没有错误）则结束熔断，否则会再次被熔断。

比如，当资源近 1 分钟的异常数目超过阈值之后会进行熔断。

注意

要注意，异常降级仅针对业务异常，对 Sentinel 限流降级本身的异常（BlockException）不生效。