ALIBABA CLOUD

阿里云

专有云企业版

应用高可用服务

用户指南

产品版本：V3.13.0

文档版本：20201225

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

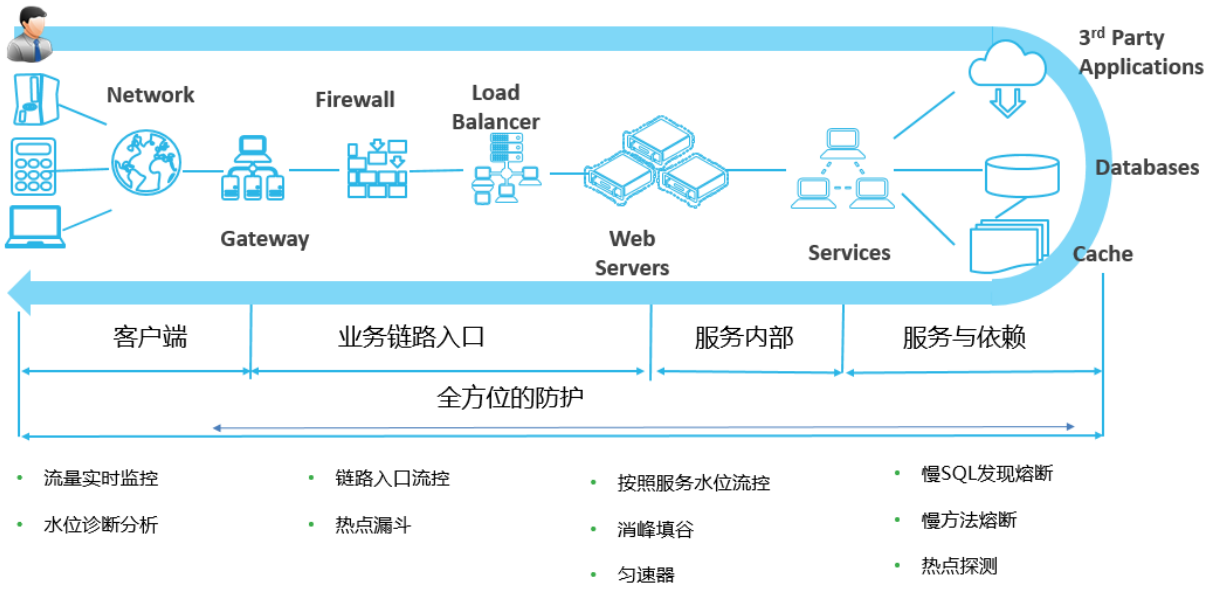
1.什么是应用高可用服务 AHAS

高可用服务 AHAS 的核心功能是流量防护即流控降级，分为应用防护和网关防护。 应用防护结合 Sentinel 的优势，以流量为切入点，从流量控制、熔断降级、系统保护等多个维度来帮助您保障服务的稳定性。网关防护是对网关进行流量控制，从流量入口处拦截激增的流量，防止下游服务被冲垮。

使用场景

AHAS 应用流控降级广泛用于秒杀场景、消息削峰填谷、集群流量控制、实时熔断等场景中，从多个维度保障您的业务稳定性。

在一个常见的分布式应用中，如下图所示。一个请求先通过终端到达 Gateway，再经过防火墙和网络负载均衡，其中还包括调用下游的其它服务和第三方应用，才能到达前端网络服务。AHAS 应用流控降级在不同的层次以流量为切面提供秒级实时的流量分析（例如在客户端层提供流量实时监控和水位诊断分析功能），帮助运维人员采取针对性的防护措施，全方位地保护应用的稳定性。



应用防护功能特性

- 秒级流量分析功能，动态规则实时推送。
- 专业多样化的防护手段：
  - 入口流量控制：按照服务容量进行流量控制，常用于应用入口，例如： Gateway、前端应用、服务提供方等。
  - 热点隔离：将热点和普通流量隔离出来，避免无效热点抢占正常流量的容量。
  - 对依赖方隔离或降级：对应用和应用之间、应用内部采用隔离或降级手段，将不稳定的依赖的对应用的影响减至最小，从而保证应用的稳定性。
  - 系统防护：AHAS 应用防护可以根据系统的能力（例如 Load、CPU 使用率等）来动态调节入口的流量，保证系统稳定性。
- 实时的单机监控能力，强大的聚合监控和历史监控查询能力。

网关防护功能特性

网关防护的主要功能如下：

- 针对路由配置中的某个路由进行流量控制，或者自定义一组 API 进行流量控制。

- 针对请求的客户端 IP、Header 或者 URL 参数进行流控。
- 限制某个 API 的调用频率，支持秒、分钟、小时、天等多个时间维度。

网关防护支持的 Spring Cloud Gateway 和 Spring Cloud Zuul 类型的应用接入。

2.快速入门

2.1. 登录 AHAS 控制台

该章节介绍了如何登录到 AHAS 的控制台。

前提条件

- 登录Apsara Uni-manager运营控制台前，确认您已从部署人员处获取Apsara Uni-manager运营控制台的服务域名地址。
- 推荐使用Chrome浏览器。

操作步骤

1. 在浏览器地址栏中，输入Apsara Uni-manager运营控制台的访问地址，按回车键。
2. 输入正确的用户名及密码。

   请向运营管理员获取登录控制台的用户名和密码。

   > **说明** 首次登录Apsara Uni-manager运营控制台时，需要修改登录用户名的密码，请按照提示完成密码修改。为提高安全性，密码长度必须为 8~20 位，且至少包含以下两种类型：
   >
   > - 英文大写或小写字母（A~Z、a~z）
   > - 阿拉伯数字（0~9）
   > - 特殊符号（感叹号（！）、at（@）、井号（#）、美元符号（$）、百分号（%）等）

3. 单击**登录**。
4. 在页面顶部的导航栏中，单击**产品**，选择 **应用高可用服务**。
5. 在**应用高可用服务**页面选择**组织**和**地域**，然后单击 **AHAS**。

2.2. 快速使用应用防护

将应用接入 AHAS 应用防护后，即可对您的应用进行流量控制、流量降级等操作。本文将使用 Demo 应用演示如何将应用接入 AHAS 并为其配置流控规则。

步骤一：接入 Demo 应用

1. 登录 AHAS 控制台。
2. 在控制台左侧导航栏选择**应用防护**。
3. 在**应用列表**页面右上角单击**新应用接入**，然后单击**体验 Demo** 页签，查看 Demo 下载地址和对应的启动命令。

i. 下载 Demo 安装包。

- 命令下载。在您的服务器中执行以下命令下载 Demo 安装包。

```
wget <Address>/sdk/latest/ahas-sentinel-sdk-demo.jar
```

> **说明** 将 <Address> 替换为您的真实环境地址，例如：ahas.server.intra.env25.shuguang.com。

- 手动下载。单击 应用防护 Demo JAR 包，手动下载安装包。

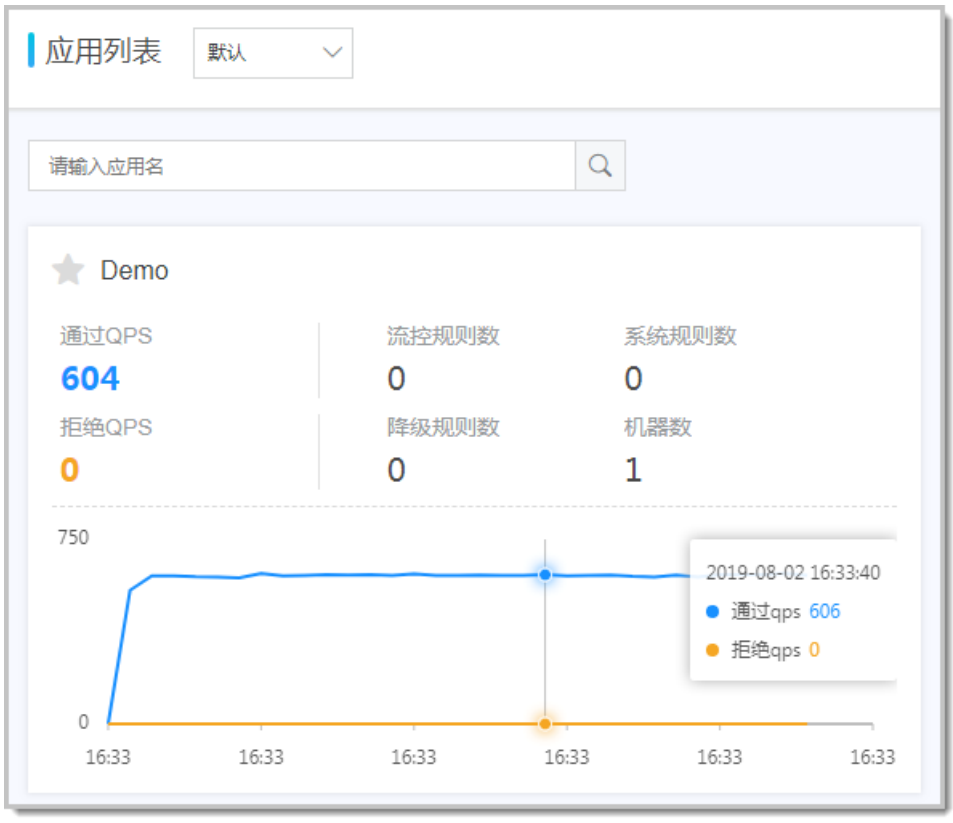
ii. 在安装包所在目录下执行以下命令启动安装包。

```
java -Dahas.namespace=default -Dproject.name=<AppName> -Dahas.gateway.address=<Address> -Daddress.server.domain=<Domain> -jar ahas-sentinel-sdk-demo.jar
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

4. 在**体验 Demo** 页签单击**我已完成上述步骤**

在**应用列表**页面，您可以看到接入 Demo 应用卡片。



步骤二：创建流控规则

本示例中，Demo 应用的 function_0 调用频繁导致系统响应时间增高，需对 Demo 应用的 function_0 资源配置 QPS 为 5 的流控规则，具体操作步骤如下：

1. 在**应用列表**页面单击应用防护 Demo 的应用卡片，进入**应用概览**页面。

2. 在控制台左侧导航栏中单击**监控详情**。
3. 在该应用的**监控详情**页面，单击 **function_0** 资源卡片右上角的加号。
4. 在**新增规则** 对话框中，选择**阈值类型**为 QPS，设置**阈值**为 5，然后单击**新建**。

结果验证

在应用的**监控详情**页面，查看配置流控规则的资源卡片。可以看到该资源的**通过 QPS** 指标从 99 降到 5。

更多信息

更多规则配置信息，请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

3.应用防护

3.1. 流控降级原则

3.1.1. 流控降级原则概述

AHAS 流控降级功能是基于 Sentinel 实现的，您在使用该功能前，请先了解以下原则。

Sentinel 原则

- Sentinel 是围绕着资源来工作的。
- 编码时，只需要关心如何定义资源，即哪些方法/代码块需要保护，而不需要关注如何保护这个资源。
- 通过添加规则来保护资源，规则添加即时生效。

规则配置原则

- 按照应用处理能力进行流控：
  - 按服务提供方流控原则，详情请参见服务提供方或消费方流控。
  - 削峰填谷原则，详情请参见削峰填谷。
  - 冷启动原则，详情请参见冷启动。
  - 联动控制原则，详情请参见关联限流。
- 强依赖隔离原则，强依赖隔离。
- 弱依赖降级原则，弱依赖降级。
- 系统保护原则，详情请参见系统防护。

3.1.2. 按应用处理能力控制流量

3.1.2.1. 服务提供方或消费方流控

AHAS 应用流控降级可以根据服务提供方的能力和服务消费方的分配能力进行流量控制。其中服务提供方（Service Provider）是指对外提供请求的服务或应用；服务消费方（Service Consumer）是指调用该服务的下游应用。

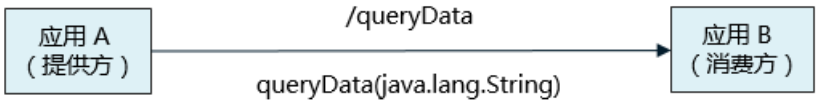
根据服务提供方限流

为了保护服务提供方不被激增的流量拖垮影响稳定性，您可以为其配置 QPS 模式的流控规则。当每秒的请求量超过设定的阈值时，AHAS 将拒绝多余的请求。提供方限流可以分为服务接口限流和服务方法限流。

- 服务接口限流：适用于整个服务接口的 QPS 不超过一定数值的情况。例如：为对应服务接口资源配置 QPS 阈值。
- 服务方法限流：适用于服务的某个方法的 QPS 不超过一定数值的情况。例如：为对应服务方法资源配置 QPS 阈值。

示例：

若应用 A 为服务提供方，其 Web 接口 `/queryData` 对应的方法是 `queryData(java.lang.String)` 。假设应用 A 最多每秒只能承受 10 次调用。若调用次数超过，则应用 A 宕机。



针对此类场景，可以对应用 A 按服务方法粒度设置流控规则，限制 `queryData(java.lang.String)` 方法每秒最多只能被调用 10 次，具体操作步骤，参见配置流控规则。若超过阈值，消费方将会收到一个 `BlockException` 异常，并且快速返回。

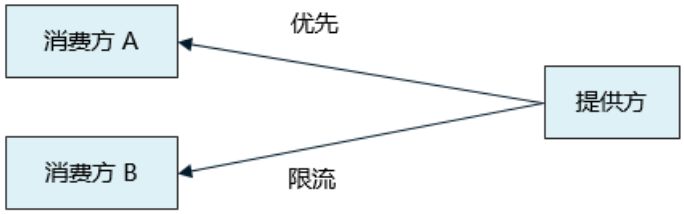
根据服务消费方限流

根据消费方限流是指根据调用方的需求来分配服务提供方的处理能力。

> **说明** 若限流规则未设置调用方（default），则该限流规则对所有调用方生效。若限流规则设置了调用方，则限流规则仅对指定调用方生效。

示例：

有两个服务消费者 A 和 B 都向服务提供方发起调用请求，我们希望优先服务消费方 A，而只对来自消费方 B 的请求进行限流。通过对消费方 B 设置限流规则（limitApp）来实现这个目的。



对于默认框架例如 Dubbo，AHAS 会自动解析 Dubbo 消费方的 `application name` 作为调用方名称（origin），在进行资源保护的时候都会带上调用方名称。而对于非默认框架，只需要在 Sentinel 原有的代码中加入 `ContextUtil.enter(resourceName, origin)` 和 `ContextUtil.exit()` 即可。示例代码如下：

```
Entry entry = null;
//这里代表消费者 A 的调用
ContextUtil.enter(queryData, "A");
try {
 // 资源名可使用任意有业务语义的字符串
 entry = SphU.entry("queryData");
 // 被保护的业务逻辑
 // do something...
} catch (BlockException e1) {
 // 资源访问阻止，被限流或被降级
 // 进行相应的处理操作
} finally {
 if (entry != null) {
  entry.exit();
 }
}
//调用结束
ContextUtil.exit();
```

> **说明** `ContextUtil.enter(xxx)` 方法仅在初次调用链路入口时才生效。

更多设置

限流规则中的 `limitApp` （针对来源）支持以下三种选项，分别对应不同的场景：

1. `default` ：适用于不区分消费者的场景。来自任何调用者的请求都将进行限流统计。如果这个资源名的调用总和超过了这条规则定义的阈值，则触发限流。
2. `{some_origin_name}` ：适用于对特定的消费者限流的场景。只有来自这个调用者的请求才会进行流量控制。

   例如，资源 A 配置了一条针对消费者 caller1 的规则，那么当且仅当来自消费者 caller1 对资源 A 的请求才会触发流量控制。
3. `other` ：表示针对除 `{some_origin_name}` 以外的其余调用方的流量进行流量控制。这个场景适用于资源对大部分消费者都有一个通用的阈值，对特定消费者有不一样的阈值的场景。

   例如，资源 A 可以对大部分消费者每秒提供 10 个请求，但是对于消费者 caller1 是个例外，对 caller1 ，每秒可以提供 200 个请求。需配置两条规则，说明如下：

   - 为消费者 caller1 配置一条 `limitApp` 为 caller1 的限流规则，这条规则每秒的最大请求量设置为 200；
   - 同时，配置一条 `limitApp` 为 other 的规则，这条规则每秒的最大请求量设置为 10，那么任意来自非 caller1 对该 resource 的调用，都不能超过 10。

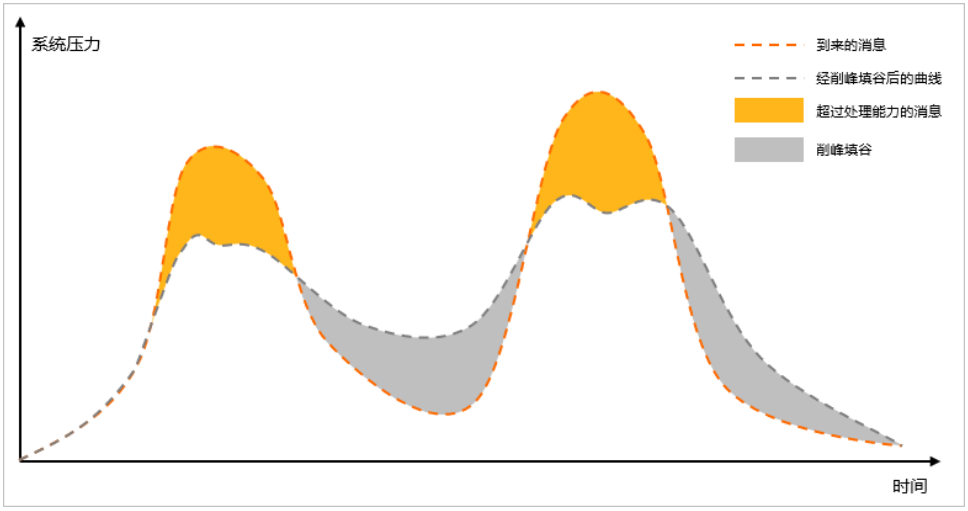
> **说明** 同一个资源名可以配置多条规则，规则的生效顺序为：{some_origin_name} > other > default。

3.1.2.2. 削峰填谷

当消费端请求骤增时，可以为其配置排队等待的流控规则，以稳定的速度逐步处理这些请求，起到“削峰填谷”的效果，从而避免流量骤增造成系统负载过高。

背景信息

在实际应用中，收到的请求是没有规律的。例如：某应用的处理请求的能力是每秒 10 个。在某一秒，突然到来了 30 个请求，而接下来两秒，都没有请求到达。在这种情况下，如果直接拒绝 20 个请求，应用在接下来的两秒就会空闲。所以，需要把骤增的请求平均到一段时间内，让系统负载保持在请求处理水位之内，同时尽可能地处理更多请求。



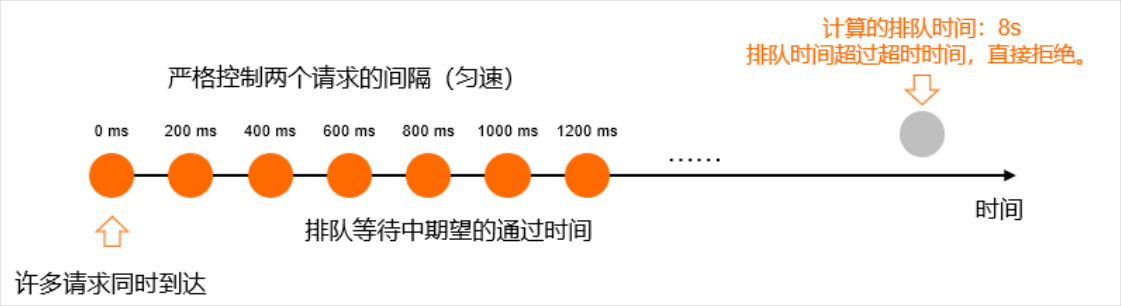
上图中，黄色的部分代表超出消息处理能力的部分。把黄色部分的消息平均到之后的空闲时间去处理，这样既可以保证系统负载处在一个稳定的水位，又可以尽可能地处理更多消息。通过配置流控规则，可以达到消息匀速处理的效果。

功能原理

AHAS 流控降级的排队等待功能，可以把骤增的大量请求匀速分配，以固定的间隔时间让请求通过，起到“削峰填谷”的效果，从而避免流量骤增造成系统负载过高的情况。堆积的请求将会被排队处理，当请求的预计排队时间超过最大超时时长时，AHAS 将拒绝这部分超时的请求。

例如：配置匀速模式下请求 QPS 为 5，则每 200 ms 处理一条请求，多余的处理任务将排队；同时设置了超时时间为 5s，则预计排队时长超过 5s 的处理任务将被直接拒绝。具体操作步骤，参见配置流控规则。

示意图如下：



3.1.2.3. 冷启动

对于长期处于低水位状态的系统，可以使用冷启动功能来避免流量骤增导致水位瞬间升高系统不可用的情况。

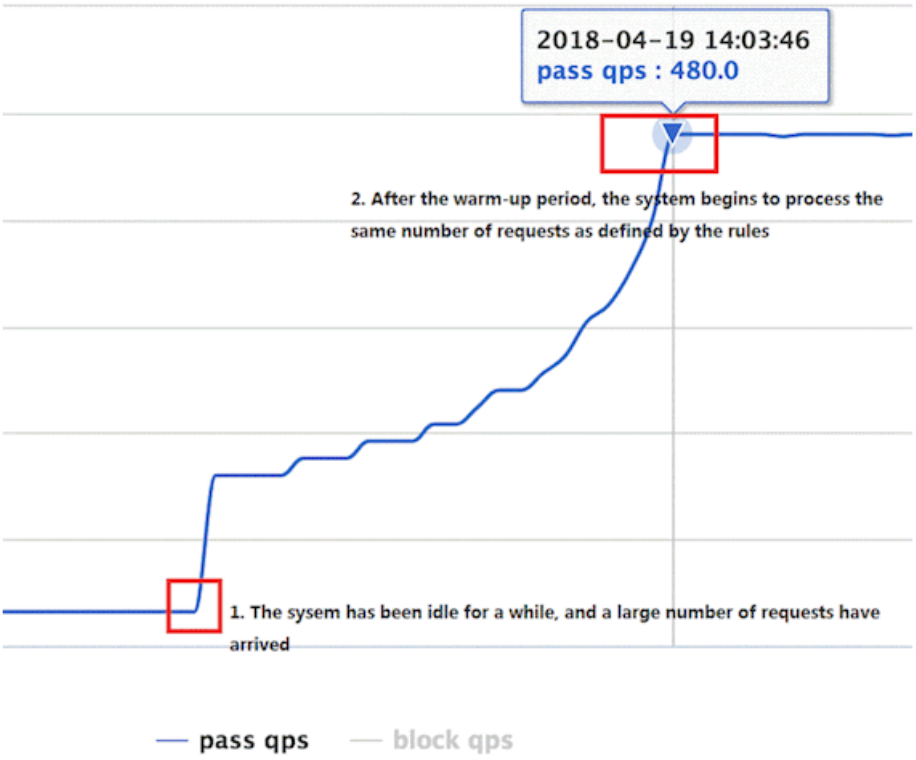
功能原理

当系统长期处于低水位的情况下，若流量突然增加，可能会把系统水位瞬间拉高把系统压垮。通过配置冷启动规则，可以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。

冷启动，参考了 Guava 的算法，通过随时调整斜率，把流量在指定的时间之内缓慢调整到特定的阈值。

示例

若对系统设置**流控模式**为直接，**流控方式**为 **Warm Up**，**预热时间**为 200 的流控规则，具体操作步骤，参见配置流控规则。设置规则后，可以看到流量的增长趋势如下图所示：



3.1.2.4. 关联限流

使用关联限流策略，可以避免具有关联关系的资源之间过度的争抢，造成的资源不可用问题。

当两个资源之间具有资源争抢或者依赖关系的时候，这两个资源便具有了关联。例如对数据库同一个字段的读操作和写操作存在争抢，读的速度过高会影响写的速度，写的速度过高会影响读的速度。如果放任读写操作争抢资源，则争抢本身带来的开销会降低整体的吞吐量。可使用关联限流来避免具有关联关系的资源之间过度的争抢。

示例

`read_db` 和 `write_db` 这两个资源分别代表数据库读写。给 `read_db` 设置以下规则来达到写优先的目的。具体操作步骤，请参见配置流控规则。

当写库操作过于频繁时，读数据的请求会被限流。`read_db` 会在 `write_db` 资源的 QPS 超过 10 之后，调用被拒绝。

3.1.3. 弱依赖降级

若依赖的第三方应用出错不会影响整体流程，则称之为弱依赖。对于弱依赖不稳定时，需要配置降级原则来保护系统稳定性。

背景信息

在实际业务中，应用通常会调用依赖方（远程服务、数据库、第三方 API 等）来完成服务。例如，支付时需要远程调用银联提供的 API。然而依赖方的稳定性是不能保证的。若依赖方出现不稳定的情况，则请求和调用依赖方的方法的响应时间变长，线程产生堆积，最终可能耗尽自身的线程数，导致应用本身不可用。

在复杂链路中，若某一环不稳定，就可能会层层渲染，最终导致整个链路都不可用。

针对以上情况，可以使用 AHAS 应用流控降级功能对依赖方配置降级原则来保证系统稳定性。

功能原理

若应用依赖于多个下游服务（弱依赖），当下游服务调用过慢，则会严重影响当前服务的调用。为调用端配置基于平均响应时间或错误率的降级规则后，当调用链路中某个下游服务调用的平均响应时间或错误率超过阈值，AHAS 就会对此调用进行降级操作，拒绝多余的调用，保护应用不被调用端短板影响。

配置降级规则具体操作步骤，请参见配置降级规则。

同时可以配合 Fallback 功能使用，在被降级的时候提供相应的处理逻辑。

3.1.4. 强依赖隔离

若依赖的第三方应用或组件，或者应用自身的内部方法出错会影响整体流程，则称之为强依赖。对于强依赖，需要配置隔离原则来保护系统稳定性。

功能原理

当强依赖出现不稳定的时候，可以通过配置并发线程数隔离原则来限制不稳定的强依赖并发数，隔离强依赖。配置并发线程数隔离原则后，无需再进行线程池隔离，AHAS 会控制资源的线程数。当请求数超过阈值时，AHAS 将拒绝多余的请求，直到堆积的线程处理完成，以此来达到信号量隔离的效果。

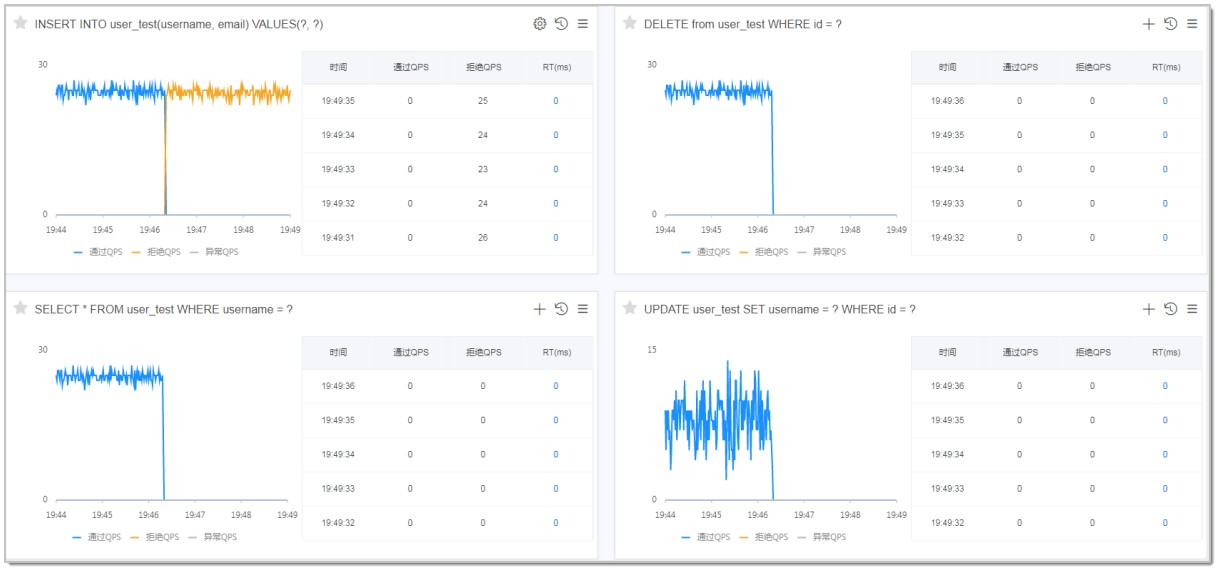
线程数目超出时，设置快速失败能够有效地防止自己被慢调用所影响。

示例

若有应用 A 的一个内部方法 `user_test` 不稳定，出现慢 SQL。

针对此类场景，可以为调用设置隔离规则（将阈值类型选择为线程数，并将阈值设置为 0），具体操作步骤，参见配置流控规则。

配置规则后， `user_test` 方法被限流，各 SQL 调用情况如下图所示。



3.1.5. 系统防护

系统防护即从整体维度对应用入口流量进行控制，结合应用的 Load、总体平均 RT、入口 QPS 和线程数等几个维度的监控指标，让系统的入口流量和系统的负载达到一个平衡，让系统尽可能跑在最大吞吐量的同时保证系统整体的稳定性。

背景信息

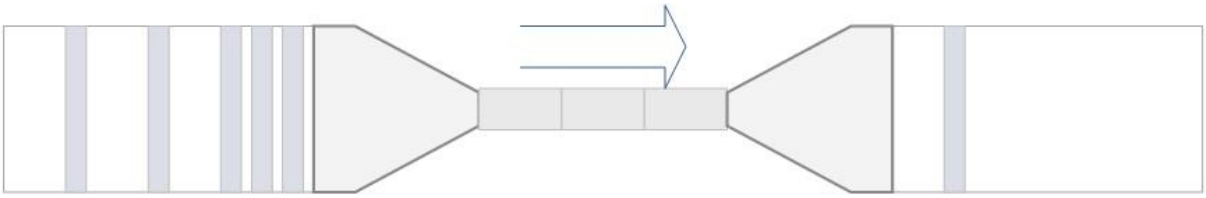
长期以来，系统自适应保护的思路是根据硬指标即系统的负载来做系统过载保护。即当系统负载高于某个阈值，就禁止或者减少流量的进入；若负载恢复，则恢复流量的进入。这样会造成两个不可避免的问题：

- 若根据负载的情况来调节流量的通过率，则会产生延迟。若当前通过率的调整会导致负载增大，那么至少要过 1 秒之后才能被观测到；同理，若当前通过率调整会使负载降低，也需要 1 秒之后才能继续调整。这种方法会浪费系统的处理能力。导致我们看到的负载曲线产生锯齿。
- 通过率恢复慢。在下游应用不可靠，应用响应时间很长，从而导致负载很高的场景中，若下游应用恢复时，应用响应时间也会随之缩短，此时通过率理应会大幅度增大。但由于此时负载仍然很高，所以通过率的恢复慢。

为解决上述问题，AHAS 应用流控降级在系统自适应保护的做法是：用每分钟的负载作为启动控制流量，使用请求的响应时间以及当前系统正在处理的请求速率来决定通过的流量。旨在在系统不被拖垮的情况下，提高系统的吞吐率。

功能原理

我们把系统处理请求的过程想象为一个水管，到来的请求是往这个水管灌水，当系统处理顺畅的时候，请求不需要排队，直接从水管中穿过，这个请求的RT是最短的；反之，当请求堆积的时候，那么处理请求的时间则会变为：排队时间 + 最短处理时间。



若用 T 来表示水管内部的水量，用 RT 来表示请求的处理时间，用 P 来表示进来的请求数，那么一个请求从进入水管道到从水管出来，这个水管会存在 `P * RT` 个请求。即当 `T ≈ QPS * Avg(RT)` 的时候，可以认为系统的处理能力和允许进入的请求个数达到了平衡，系统的负载不会继续增加。当入口的流量是水管出来的流量的最大的值的时候，水管的处理能力达到最大利用。

系统规则

系统保护规则是应用整体维度的，而不是资源维度的，并且仅对入口流量生效。入口流量指的是进入应用的流量（EntryType.IN），例如 Web 服务或 Dubbo 服务端接收的请求，都属于入口流量。系统规则支持四种阈值类型：

- Load（仅对 Linux/Unix-like 机器生效）：当系统 load1 超过阈值且系统当前的并发线程数超过系统容量时才会触发系统保护。
- RT：当单台机器上所有入口流量的平均 RT 达到阈值即触发系统保护。
- 线程数：当单台机器上所有入口流量的并发线程数达到阈值即触发系统保护。
- 入口 QPS：当单台机器上所有入口流量的 QPS 达到阈值即触发系统保护。

为应用配置系统规则，具体操作步骤请参见配置系统规则。

3.2. 接入应用

3.2.1. 接入 Spring Boot 应用

将 Spring Boot 应用接入 AHAS 应用防护后，可以对其配置流控、降级和系统规则来保证系统稳定性。本文将帮助您了解如何使用 SDK 方式将 Spring Boot 应用接入应用防护。

操作步骤

1. 登录 AHAS 控制台。
2. 在左侧导航栏，单击**应用防护**。
3. 在**应用列表**页面右上角单击**新应用接入**，然后单击**Spring Boot 应用接入**页签。
4. 在 Spring Boot 应用的 Pom 文件中引入依赖：

```
<dependency>
 <groupId>com.alibaba.csp</groupId>
 <artifactId>spring-boot-starter-ahas-sentinel-client</artifactId>
 <!-- 可指定版本号，最新版本见 AHAS 控制台>应用防护> Springboot 应用接入页。 -->
 <version>x.y.z</version>
</dependency>
```

> **说明** 在Spring Boot 应用接入页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

5. 在应用工程中添加埋点
   - 添加 HTTP 埋点： 引入 spring-boot-starter-ahas-sentinel-client 依赖后，应用会自动添加 Web 接口埋点。
   - 添加自定义埋点：
     a. 引入spring-boot-starter-aop。

     > **说明** 若您的工程中已引入此依赖，则跳过此步骤。

     b. 在业务方法使用注解作为埋点。

     若不配置 BlockHandler，则被流控降级时方法会直接抛出 BlockException，若方法未定义 Throws BlockException 则会被 JVM 包装一层 UndeclaredThrowableException。BlockHandler 和 Fallback 函数的方法签名有限制，详情请参见*《AHAS 应用高可用服务开发指南》*中**配置触发规则后的逻辑**章节。

```
@SentinelResource(value = "getUserById")
public User getUserById(String id) {
 return new User(id);
}
```

> **说明** 若您从 1.5.0-apsara 之前的版本升级到 1.5.0-apsara，或自己额外引入了 Web filter 等的 Bean，需要先将之前注册 Bean 的相关代码去掉，否则可能会导致重复统计。

6. 配置应用的启动参数。

```
ahas.namespace=default
ahas.gateway.address=ahas.gateway.env49.cloud.arm.com:9527
ahas.address.server.domain=jmenv-dncs.cn-hangzhou-arm-amtest96001-a.env49.cloud.arm.com
project.name=AppName
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

7. （可选）您可以自定义 Spring Boot 应用触发限流、降级或系统保护规则时的处理逻辑。详情请参见《*AHAS 应用高可用服务开发指南*》中**配置触发规则后的逻辑**章节。

   > **说明** 若未执行此步骤，当 Spring Boot 应用触发流控降级规则时，默认抛出 `BlockException` 异常类的子类（触发流控规则，则抛出流控异常 `FlowException`；触发降级规则，则抛出降级异常 `DegradeException`）。

8. 重新启动您的应用。

结果验证

启动应用并调用配置埋点的方法。若该应用出现在 AHAS 控制台**应用列表**页面，且在该应用的**监控详情**页面有能看到配置埋点的方法，则说明接入成功。

后续操作

为应用配置流控降级规则请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

3.2.2. 接入 Spring 应用接入

将 Spring 应用接入 AHAS 应用防护后，可以对其配置流控、降级和系统规则来保证系统稳定性。本文将帮助您了解如何使用 SDK 方式将普通 Spring 应用接入应用防护。

操作步骤

1. 登录 AHAS 控制台。
2. 在左侧导航栏，单击**应用防护**。
3. 在**应用列表**页面右上角单击**新应用接入**，然后单击**Spring 应用接入**页签。
4. 在 Spring 应用的 Pom 文件中引入依赖。

```
<dependency>
 <groupId>com.alibaba.csp</groupId>
 <artifactId>ahas-sentinel-client</artifactId>
 <!-- 可指定版本号，最新版本见 AHAS 控制台>应用防护> Springboot 应用接入页。 -->
 <version>x.y.z</version>
</dependency>
```

> **说明** 在Spring 应用接入页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

5. 在应用工程中添加埋点。
   - 添加 HTTP 埋点：引入 spring-boot-starter-ahas-sentinel-client 依赖后，应用会自动添加 Web 接

- 添加 HTTP 埋点：引入 spring-boot-starter-ahas-sentinel-client 依赖后，应用会自动添加 Web 接口埋点。
- 添加普通接口埋点：

  a. 引入spring-boot-starter-aop。

  > **说明** 若您的工程中已引入此依赖，则跳过此步骤。

  b. 在业务方法使用注解作为埋点。

  若不配置 BlockHandler，则被流控降级时方法会直接抛出 BlockException，若方法未定义 Throws BlockException 则会被 JVM 包装一层 UndeclaredThrowableException。BlockHandler 和 Fallback 函数的方法签名有限制，详情请参见*《AHAS 应用高可用服务开发指南》*中**配置触发规则后的逻辑**章节。

```
@SentinelResource(value = "getUserById")
public User getUserById(String id) {
 return new User(id);
}
```

> **说明**
>
> - 若您从 1.5.0-apsara 之前的版本升级到 1.5.0-apsara，或自己额外引入了 Web filter 等的 Bean，需要先将之前注册 Bean 的相关代码去掉，否则可能会导致重复统计。
> - 老版本spring框架并不支持使用注解的方式声明 Filter 或是 SentinelResourceAspect，如果版本过低时，用 xml 方式声明即可。

6. 配置应用的启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> **说明** 普通spring 应用必须通过指定 -D 参数的方式来配置来配置启动参数。<AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

7. （可选）您可以自定义 Spring Boot 应用触发限流、降级或系统保护规则时的处理逻辑。 详情请参见*《AHAS 应用高可用服务开发指南》*中**配置触发规则后的逻辑**章节。

> **说明** 若未执行此步骤，当 Spring 应用触发流控降级规则时，默认抛出 `BlockException` 异常类的子类（触发流控规则，则抛出流控异常 `FlowException`；触发降级规则，则抛出降级异常 `DegradeException`）。

8. 重新启动您的应用。

结果验证

启动应用并调用配置埋点的方法。若该应用出现在 AHAS 控制台**应用列表**页面，且在该应用的**监控详情**页面有能看到配置埋点的方法，则说明接入成功。

后续操作

为应用配置流控降级规则请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

3.2.3. 接入 Dubbo 应用

将 Dubbo 应用接入 AHAS 应用防护后，可以对其配置流控、降级和系统规则来保证系统稳定性。本文将帮助您了解如何使用 SDK 方式将 Dubbo 应用接入应用防护。

操作步骤

1. 登录 AHAS 控制台。
2. 在左侧导航栏，单击**应用防护**。
3. 在**应用列表**页面右上角单击**新应用接入**，然后单击 **Dubbo 应用接入**页签。
4. 选择以下任意一种方式，在 Dubbo 应用中添加应用防护依赖。
   - 在 Dubbo 应用的 Pom 文件中添加以下依赖：

```
<dependency>
 <groupId>com.alibaba.csp</groupId>
 <artifactId>ahas-sentinel-client</artifactId>
 <!-- 可指定版本号，最新版本见 AHAS 控制台应用防护应用接入页引导。 -->
 <version>x.y.z</version>
</dependency>
```

> **说明** 在 **Dubbo 应用接入**页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

若您使用 Dubbo 2.7.x 版本，请引入以下依赖：

```
<!-- ahas-sentinel-client 使用 1.5.0 版本 --><dependency>
<groupId>com.alibaba.csp</groupId>
<artifactId>ahas-sentinel-client</artifactId>
<version>1.5.0-apsara</version>
 <exclusions>
  <exclusion>
   <groupId>com.alibaba.csp</groupId>
   <artifactId>sentinel-dubbo-adapter</artifactId>
  </exclusion>
 </exclusions>
</dependency>
<dependency>
 <groupId>com.alibaba.csp</groupId>
 <artifactId>sentinel-apache-dubbo-adapter</artifactId>
 <version>1.6.3</version>
</dependency>
```

- 添加 JAR 包依赖。

  在**Dubbo 应用接入**页面单击**请点此链接下载**，并将下载的压缩包中的所有 JAR 包放置在 *classpath* 目录下。

5. 配置启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

6. （可选）您可以自定义 Dubbo 应用触发限流、降级或系统保护规则时的 fallback 处理逻辑，自定义 `DubboFallback` 接口并通过 `DubboFallbackRegistry` 注册即可。配置后，当 Dubbo 应用触发流控、降级或系统规则时，AHAS 会将 `BlockException` 包装后抛出。详情请参见 *《AHAS 应用高可用服务开发指南》***配置触发规则后的逻辑**章节。

> **说明** 若未执行此步骤，当 Dubbo 应用触发流控降级规则时，默认抛出 `BlockException` 异常类的子类（触发流控规则，则抛出流控异常 `FlowException`；触发降级规则，则抛出降级异常 `DegradeException`）。

7. 重新启动您的应用。

结果验证

启动应用并调用配置埋点的方法。若该应用出现在 AHAS 控制台**应用列表**页面，且在该应用的**监控详情**页面有能看到配置埋点的方法，则说明接入成功。

后续操作

为应用配置流控降级规则请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

3.2.4. 接入 Web 应用

将 Web 应用接入 AHAS 应用防护后，可以对其配置流控、降级和系统规则来保证系统稳定性。本文将帮助您了解如何使用 SDK 方式将 Web 应用接入应用防护。

操作步骤

1. 登录 AHAS 控制台。
2. 在左侧导航栏，单击**应用防护**。
3. 在**应用列表**页面右上角单击**新应用接入**，然后单击 **Web 应用接入**页签。
4. 选择以下任意一种方式，在 Web 应用中添加应用流控降级依赖。
   - 在 Web 应用的 Pom 文件中添加以下依赖：

     ```
     <dependency>
      <groupId>com.alibaba.csp</groupId>
      <artifactId>ahas-sentinel-client</artifactId>
      <version>x.y.z</version>
     </dependency>
     ```

     > **说明** 在**Web 应用接入**页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

   - 添加 JAR 包依赖。

     在**Web 应用接入**页面单击**请点此链接下载**，并将下载的压缩包中的所有 JAR 包放置在 *classpath* 目录下。
5. 在 web.xml 文件中引入拦截器，并将其配置为第一个 Filter，示例如下：

```
<filter>
  <filter-name>SentinelCommonFilter</filter-name>
  <filter-class>com.alibaba.csp.sentinel.adapter.servlet.CommonFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>SentinelCommonFilter</filter-name>
<!-- 配置要拦截的 URL 模式 -->
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

6. 配置应用的启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

7. （可选）您可以通过以下两种方式来设定自定义的跳转 URL，当请求触发流控、降级或系统规则时会自动跳转至设定的 URL。
   - 通过 `WebServletConfig.setBlockPage(blockPage)` 方法设定自定义的跳转 URL。
   - 通过 UrlBlockHandler 接口编写定制化的限流处理逻辑，然后将其注册至 WebCallbackManager 中来实现，详情请参见《*AHAS 应用高可用服务开发指南*》中**扩展接口**章节。

   > **说明** 默认情况下，当请求触发流控、降级或系统规则时会返回默认提示页面，提示信息为：`Blocked by Sentinel (flow limiting)`。

8. （可选）可以自定义 URL 清理，防止 `/payment/{id}` 这类 REST 风格的 URL 堆积过多影响系统性能，配置 URL 清理的方式有两种：
   - 实现自定义的 UrlCleaner 接口，自定义 URL 清理归一处理逻辑，详情请参见《*AHAS 应用高可用服务开发指南*》中**扩展接口**章节。
   - 通过 *properties* 文件配置前缀清理。

     将 *properties* 文件中 URL 格式更改为：匹配前缀=清理后的资源名称，例如 /payment/=/payment/*。再通过启动参数 `-Dcsp.sentinel.url.clean.config.path` 配置文件路径。

     > **说明** 启动参数中的默认路径为 JAR 包所在的 ahas-sentinel-url-clean.properties 文件。启动参数支持 classpath 形式，例如：`Dcsp.sentinel.url.clean.config.path=classpath:ahas-sentinel-url-clean.properties`。

9. 重新启动您的应用。

结果验证

启动应用并调用配置埋点的方法。若应用出现在 AHAS 控制台**应用列表**页面，且在该应用的**监控详情**页面有能看到配置埋点的方法，则说明接入成功。

后续操作

为应用配置流控降级规则请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

3.2.5. 接入 MyBatis 应用

将 MyBatis 应用接入 AHAS 应用防护后，可以对其配置流控、降级和系统规则来保证系统稳定性。本文将帮助您了解如何使用 SDK 方式将 MyBatis 应用接入应用防护。

操作步骤

1. 通过以下任意一种方式，为应用添加依赖。
   - 方式一：在 Pom 文件中添加依赖：

     ```
     <dependency>
      <groupId>com.alibaba.csp</groupId>
      <artifactId>ahas-sentinel-client</artifactId>
      <version>x.y.x</version>
     </dependency>
     ```

     > 说明 在 AHAS 控制台**应用防护 > 新应用接入 > Springboot 应用接入**页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

   - 方式二：添加 JAR 包依赖。
     a. 单击下载 ahas-sentinel 依赖包 。
     b. 解压依赖包，并将依赖包中的所有 JAR 包放置在 *classpath* 下。
2. 添加拦截器。
   - 若您使用了 MyBatis Spring Boot Starter，您可以直接将 SentinelMyBatisInterceptor 注册为 bean：

     ```
     @Configuration
     public class MyBatisConfig {
       @Bean
       public SentinelMyBatisInterceptor sentinelMyBatisInterceptor() {
         return new SentinelMyBatisInterceptor();
       }
     }
     ```

   - 若您未使用 MyBatis Spring Boot Starter ，则需在 MyBatis 应用的 XML 配置文件中引入

SentinelMyBatisInterceptor 拦截器依赖：

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
    PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
  <plugins>
   <!-- 引入 AHAS Sentinel 拦截器 -->
    <plugin interceptor="com.alibaba.csp.sentinel.demo.db.mybatis.interceptor.SentinelMyBatisInterceptor"/>
  </plugins>
</configuration>
```

3. 通过以下任意一种方式，配置应用的启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

4. 重新启动您的应用。

结果验证

启动应用并调用配置埋点的方法。若应用出现在 AHAS 控制台**应用列表**页面，且在该应用的**监控详情**页面有能看到配置埋点的方法，则说明接入成功。

后续操作

为应用配置流控降级规则请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

当 MyBatis 应用触发配置的流控、降级或系统保护规则时，会抛出 `MyBatisSentinelBlockException` 类型的异常，您可以自行捕获该异常并进行处理。

3.2.6. 通过自定义埋点接入

通过自定义 Java SDK 埋点的方式将应用接入 AHAS 应用防护，可以更加灵活地对任意代码块进行操作。本文介绍如何通过自定义埋点接入 AHAS 应用防护。

操作步骤

1. 登录 AHAS 控制台。
2. 在左侧导航栏中，单击**应用防护**。
3. 在**应用列表**页面右上角单击**新应用接入**，然后单击 **自定义埋点**页签。
4. 选择以下任意一种方式，在应用中添加应用流控降级依赖。
   - 在应用的 Pom 文件中添加以下依赖：

     ```
     <dependency>
      <groupId>com.alibaba.csp</groupId>
      <artifactId>ahas-sentinel-client</artifactId>
      <!-- 可指定版本号，最新版本见 AHAS 控制台>应用防护>自定义埋点页。 -->
      <version>x.y.z</version>
     </dependency>
     ```

     > **说明** 在**自定义埋点**页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

   - 添加 JAR 包依赖。

     在**自定义埋点**页面单击**请点此链接下载**，并将下载的压缩包中的所有 JAR 包放置在 *classpath* 目录下。
5. 使用以下代码包住您的业务逻辑：

   ```
   Entry entry = null;
   try {
    // HelloWorld 为该调用资源名称，控制台上监控/簇点链路会显示该资源
    // 配置规则的时候也是针对某个资源名称配置
    // EntryType.IN 标识该资源为入口资源，系统规则只针对入口资源生效
    entry = SphU.entry("HelloWorld", EntryType.IN);
    // 被保护的业务逻辑
    System.out.println("Hello Sentinel!");
   } catch (BlockException e) {
    // 触发流控降级，在此处进行限流处理（如 fallback 或记录日志）
   } finally {
    // 请确保 exit() 逻辑在此处被调用，并且和 SphU.entry() 成对出现
    if (entry != null) {
     entry.exit();
    }
   }
   ```

> 说明
> - EntryType 用于区分该资源是入口流量（inbound）还是出口流量（outbound），EntryType.IN 代表入口流量，EntryType.OUT 代表出口流量。系统规则只对入口资源生效。
> - 控制台展示的监控的应用总流量仅统计 EntryType.IN 的流量。
> - SphU 相关 API 文档请参见《AHAS 应用高可用服务开发指南》中**常用类及其方法**章节。

6. 配置应用的启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> 说明 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

7. （可选）使用注解方式配置应用触发限流、降级或系统保护规则时的处理逻辑。详情请参见《AHAS 应用高可用服务开发指南》中**配置触发规则后的逻辑**章节。

> 说明 若未执行此步骤，当应用触发流控降级规则时，默认抛出 BlockException 异常类的子类（触发流控规则，则抛出流控异常 FlowException；触发降级规则，则抛出降级异常 DegradeException）。

8. 重新启动您的应用。

结果验证

启动应用并调用配置埋点的方法。若应用出现在 AHAS 控制台**应用列表**页面，且在该应用的**监控详情**页面有能看到配置埋点的方法，则说明接入成功。

后续操作

为应用配置流控降级规则请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

3.2.7. 通过注解接入

通过在业务逻辑上添加注解的方式将应用接入应用防护，可以对调用方法进行注解埋点，减小对代码的入侵。本文将介绍如何添加注解将应用接入 AHAS 应用防护。

操作步骤

1. 通过以下任意一种方式，为应用添加依赖。
   - 方式一：在 Pom 文件中添加依赖：

```
<dependency>
 <groupId>com.alibaba.csp</groupId>
 <artifactId>ahas-sentinel-client</artifactId>
 <version>x.y.x</version>
</dependency>
```

> **说明** 在 AHAS 控制台**应用防护 > 新应用接入 > Spring Boot 应用接入**页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

- 方式二：添加 JAR 包依赖。
  a. 单击下载 ahas-sentinel 依赖包 。
  b. 解压依赖包，将依赖包中的所有 JAR 包放置在 *classpath* 下。

2. 引入 Spring AOP 的相关依赖。
   - 非 Spring Boot 应用需引入 spring-aop 依赖，选择版本后可查看对应的引入代码示例。
   - Spring Boot 或 Spring Cloud 应用需引入 spring-boot-starter-aop 依赖，选择版本后可查看对应的引入代码示例。
3. 将 SentinelResourceAspect 注册为一个 Spring Bean：

```
@Configuration
public class SentinelAspectConfiguration {
 @Bean
 public SentinelResourceAspect sentinelResourceAspect() {
  return new SentinelResourceAspect();
 }
}
```

4. 在业务方法上添加 @SentinelResource 注解：

```
// 原本的业务方法
@SentinelResource(value = "getUserById")
public User getUserById(String id) {
 return new User(id);
}
```

5. 通过以下任意一种方式，配置应用的启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

6. （可选）使用注解方式配置应用触发限流、降级或系统保护规则时的处理逻辑。详情请参见《*AHAS 应用高可用服务开发指南*》中**配置触发规则后的逻辑**章节。

> **说明** 若未执行此步骤，当应用触发流控降级规则时，默认抛出 `BlockException` 异常类的子类（触发流控规则，则抛出流控异常 `FlowException`；触发降级规则，则抛出降级异常 `DegradeException`）。

结果验证

启动应用并调用配置埋点的方法。若应用出现在 AHAS 控制台**应用列表**页面，且在该应用的**监控详情**页面能看到配置埋点的方法，则说明接入成功。

后续操作

为应用配置流控降级规则请参见以下文档：

- 配置流控规则
- 配置降级规则
- 配置系统规则

3.2.8. 通过开源 Sentinel SDK 接入

通过开源组件 Sentinel，可以便捷地接入 AHAS 应用防护。

操作步骤

使用开源组件 Sentinel 将应用接入 AHAS 应用防护请参见新手指南 来连接 AHAS 应用防护控制台。

若您已接入 Sentinel 控制台，您可以将 Pom 包中的 `sentinel-transport-simple-http` 模块替换为 `ahas-sentinel-client` 模块，接入 AHAS 应用防护。

> **注意** 若在本机或非阿里云 VPC 网络运行，请注意在 AHAS 控制台左上角选择地域为公网。

3.3. 配置规则

3.3.1. 配置流控规则

配置流控规则的原理是监控应用或服务流量的 QPS 指标，当指标达到设定的阈值时立即拦截流量，避免应用被瞬时的流量高峰冲垮，从而保障应用高可用性。本文介绍如何配置和管理流控规则。

前提条件

将应用接入 AHAS 应用防护，具体操作请参见以下文档：

- 接入 Spring Boot 应用
- 接入 Spring 应用接入

- 接入 Dubbo 应用
- 接入 Web 应用
- 接入 MyBatis 应用
- 通过自定义埋点接入
- 通过注解接入
- 通过开源 Sentinel SDK 接入

背景信息

流量控制在网络传输中是一个常用的概念，常用于调整网络包的发送数据。系统需处理的请求是随机不可控的，而系统的处理能力是有限的，因此就需要根据系统的处理能力对流量进行控制。流控规则的配置通常有以下场景：

- 服务提供方或消费方流控
- 削峰填谷
- 冷启动
- 关联限流

新建流控规则

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**应用防护**。
3. 在**应用列表**页面单击目标应用卡片。
4. 选择以下任意一种方法进入新建流控规则页面：
   - 在左侧导航栏单击**监控详情**，在**监控详情**页面单击资源卡片右上角加号图标 ，然后在**新增规则**对话框中选择**流控规则**页签。
   - 在左侧导航栏单击**簇点链路**，然后单击目标资源**操作**列中的**流控**。
   - 在左侧导航栏单击**流控规则**页签，然后在页面右上角单击**新增流控规则**。
5. 在**新建流控规则**对话框中配置规则信息。

新建流控规则

- **资源名称**：即待流控的资源名称。
- **来源应用**：即该规则针对的来源应用，默认来源应用设为 `default`，代表不区分来源应用。
- **阈值类型**：可选择 QPS 或线程数。
  - **QPS**：应用或服务流量的 QPS 指标。
  - **线程数**：资源的并发线程数（即该资源正在执行的线程数）阈值。
- **流控模式**：选择资源调用关系进行流控。资源关系有：**直接**、**关联**和**链路**三种。
  - **直接**：直接控制来自**流控应用**中调用来源的访问流量，如果**流控应用**为 `default` 则不区分调用来源。
  - **关联**：控制当前资源的关联资源的流量。 详情请参见关联限流。

    例如，`read_db` 和 `write_db` 这两个资源分别代表数据库读写。可以给 `read_db` 设置限流规则，设置**关联资源** 为 `write_db` 。这样当写库操作过于频繁时，读数据的请求会被限流。
  - **链路**：控制该资源所在的调用链路的入口流量。选择**链路**后需要继续配置**入口资源**，即该调用链路入口的上下文名称。
- **流控方式**：当阈值类型**阈值类型**为 QPS 时，可以选择流控方式来处理被拦截的流量。
  - **快速失败**：达到阈值时，立即拦截请求。

- Warm Up需设置具体的预热时间。详情请参见冷启动。

  如果系统在此之前长期处于空闲的状态，当流量突然增大的时候，该方式会让处理请求的速率缓慢增加，经过设置的预热时间以后，到达系统处理请求速率的设定值。默认会从设置的 QPS 阈值的 1/3 开始慢慢往上增加至 QPS 设置值。

- **排队等待**：请求匀速通过，允许排队等待，通常用于消息队列削峰填谷等场景。需设置具体的超时时间，达到超时时间后请求会失败。 详情请参见削峰填谷。

  例如，QPS 配置为 5，则代表请求每 200 ms 才能通过一个，多出的请求将排队等待通过。超时时间代表最大排队时间，超出最大排队时间的请求将会直接被拒绝。

- **是否开启**：打开开关表示启用该规则，关闭开关表示禁用该规则。

6. 单击**新建**。

管理流控规则

在**流控规则**页面，您可以启用、禁用、编辑或删除流控规则。

- 单流控规则启用/禁用：

  在**流控规则**页面，找到目标资源下对应的流控规则，单击状态**状态**栏的启用开关，可快速启用或禁用该规则。

- 多流控规则批量启用/禁用：

  在**流控规则**页面，勾选多个流控规则，单击**批量启用**或**批量禁用**，可快速启用或禁用多个规则。

- 编辑规则：

  在**流控规则**页面，找到目标资源下对应的流控规则，单击**操作**栏的**编辑**，可修改该规则的相关信息。

- 删除规则：

  在**流控规则**页面，找到目标资源下对应的流控规则，单击**操作**栏的删除**删除**。

3.3.2. 配置降级规则

AHAS 的降级功能可以监控应用下游依赖的响应时间或异常比例，当达到指定的阈值时立即降低下游依赖的优先级，避免应用受到影响，从而保障应用高可用性。

前提条件

将应用接入 AHAS 应用防护，具体操作请参见以下文档：

- 接入 Spring Boot 应用
- 接入 Spring 应用接入
- 接入 Dubbo 应用
- 接入 Web 应用
- 接入 MyBatis 应用
- 通过自定义埋点接入
- 通过注解接入
- 通过开源 Sentinel SDK 接入

背景信息

除了流量控制以外，对调用链路中不稳定的资源进行熔断降级也是重要措施之一。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，最终会导致请求发生堆积。熔断降级功能会在调用链路中某个资源出现不稳定时（例如某资源出现 Timeout 或异常比例升高），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错误。

降级规则配置通常用于弱依赖降级场景。

新建降级规则

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**应用防护**。
3. 在**应用列表**页面单击目标应用卡片。
4. 选择以下任意一种方法进入新建流控规则页面：
   - 在左侧导航栏单击**监控详情**，在**监控详情**页面单击资源卡片右上角加号图标，然后在**新增规则**对话框中选择流控规则**降级规则**页签。
   - 在左侧导航栏单击**簇点链路**，然后单击目标资源**操作**列中的**降级**。
   - 在左侧导航栏单击**降级规则**，然后在页面右上角单击**新增降级规则**。
5. 在**新建降级规则**对话框中配置规则信息。

   - **资源名称**：适用该规则的应用资源。
   - **阈值类型**：选择以 **RT**（响应时间）或**异常比例**作为阈值。

- 选择以**RT**（响应时间）作为阈值，并填写以毫秒为单位的时间。

  规则开启后，按照**秒级的平均响应时间**进行降级。如果持续进入 5 个请求，对应的平均 RT 都持续超过降级阈值，则会被自动降级。

- 选择以**异常比例**作为阈值，并在**比例**字段中填写 0 到 1 之间的小数作为比例。

  规则开启后，按照**秒级的异常比例**进行降级。当资源的每秒异常总数占通过量的比值超过阈值之后，资源进入降级状态。

- **时间窗口**：即降级触发后持续的时间。资源进入降级状态后，在配置的降级窗口时间内，请求都会快速失败。
- **是否开启**：打开表示启用该规则，关闭表示禁用该规则。

6. 单击**新建**。

管理降级规则

在**降级规则**页面，您可以启用、禁用、编辑或删除降级规则。

- 单降级规则启用/禁用：

  在**降级规则**页面，找到目标资源下对应的降级规则，单击**状态**栏的启用开关，可快速启用或禁用该规则。

- 多个降级规则批量启用/禁用：

  在**降级规则**页面，勾选多个降级规则，单击**批量启用**或**批量禁用**，可快速启用或禁用多个规则。

- 编辑规则：

  在**降级规则**页面，找到目标资源下对应的降级规则，单击**操作**栏的**编辑**，可修改该规则的相关信息。

- 删除规则：

  在**降级规则**页面，找到目标资源下对应的降级规则，单击**操作**栏的**删除**。

3.3.3. 配置系统规则

系统规则从整体维度对应用入口流量进行控制，结合应用的 Load、CPU 使用率、总体平均 RT、入口 QPS 和并发线程数等几个维度的监控指标，结合自适应的流控策略，让系统的入口流量和系统的负载达到一个平衡，保证系统在最大吞吐量状态下稳定运行。

前提条件

将应用接入 AHAS 应用防护，具体操作请参见以下文档：

- 接入 Spring Boot 应用
- 接入 Spring 应用接入
- 接入 Dubbo 应用
- 接入 Web 应用
- 接入 MyBatis 应用
- 通过自定义埋点接入
- 通过注解接入
- 通过开源 Sentinel SDK 接入

背景信息

系统保护规则是应用整体维度的，而不是资源维度的，并且仅对入口流量生效。入口流量指的是进入应用的流量，例如 Web 服务或 Dubbo 服务端接收的请求。通过自定义埋点接入和通过注解接入的应用，入口流量为 EntryType 为 `EntryType.IN` 的逻辑被调用时产生的流量。系统规则支持以下的模式：

- Load（仅对 Linux/Unix-like 机器生效）：当系统 Load1 超过阈值且系统当前的并发线程数超过系统容量时才会触发系统保护。
- CPU 使用率：当系统 CPU 使用率超过阈值（0.0~1.0）即触发系统保护。
- RT：当单台机器上所有入口流量的平均 RT 达到阈值即触发系统保护。
- 线程数：当单台机器上所有入口流量的并发线程数达到阈值即触发系统保护。
- 入口 QPS：当单台机器上所有入口流量的 QPS 达到阈值即触发系统保护。

对于一个应用来说，每种相同的系统保护规则最多只能存在一条，即一个应用最多配置五个系统保护规则。

新建系统规则

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**应用防护**。
3. 在**应用列表**页面单击目标应用卡片。
4. 在左侧导航栏单击**规则管理**，然后单击**系统规则**页签。
5. 在**系统规则**页签右上角单击**新建系统保护规则**。
6. 在**新建系统保护规则**对话框中，配置规则信息。

新增系统保护规则

- **CPU 使用率**：当系统 CPU 使用率超过阈值即触发系统保护，阈值设置范围为 0.0~1.0（代表 0%~100%）。
- **Load**：当系统的 Load1 超过阈值，且系统当前的并发线程数超过系统容量时才会触发系统保护。系统容量由系统的 maxQps * minRt 计算得出。
- **RT**：当单台机器上所有入口流量的平均 RT 达到阈值即触发系统保护，单位是毫秒。
- **线程数**：当单台机器上所有入口流量的并发线程数达到阈值即触发系统保护。
- **入口 QPS**：当单台机器上所有入口流量的 QPS 达到阈值即触发系统保护。
- **是否开启**：打开表示启用该规则，关闭表示禁用该规则。

7. 单击**新增**。

管理系统规则

在**系统规则**页签中，您可以启用、禁用、编辑或删除系统规则。

- 单系统规则启用、禁用：

  在**系统规则**页签中，找到目标应用下对应的系统规则，单击**状态**栏的启用开关，可快速启用或禁用该规则。

- 多个系统规则批量启用、禁用：

  在**系统规则**页签中，勾选多个系统规则，单击**批量启用**或**批量禁用**，可快速启用或禁用多个规则。

- 编辑规则：

  在**系统规则**页签中，找到目标应用下对应的系统规则，单击**操作**栏的**编辑**，可修改该规则的阈值。**阈值类型**不允许更改。

- 删除规则：

  在**系统规则**页签中，找到目标资源下对应的系统规则，单击**操作**栏的**删除**。

3.4. 管理应用

3.4.1. 应用概览

将应用接入 AHAS 应用防护后，AHAS 将监控应用和资源 API 维度的实时数据，从而评估系统的整体表现，并为流控降级规则的配置提供重要依据。

功能入口

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**应用防护**。
3. 在**应用列表**页面单击目标应用的资源卡片。

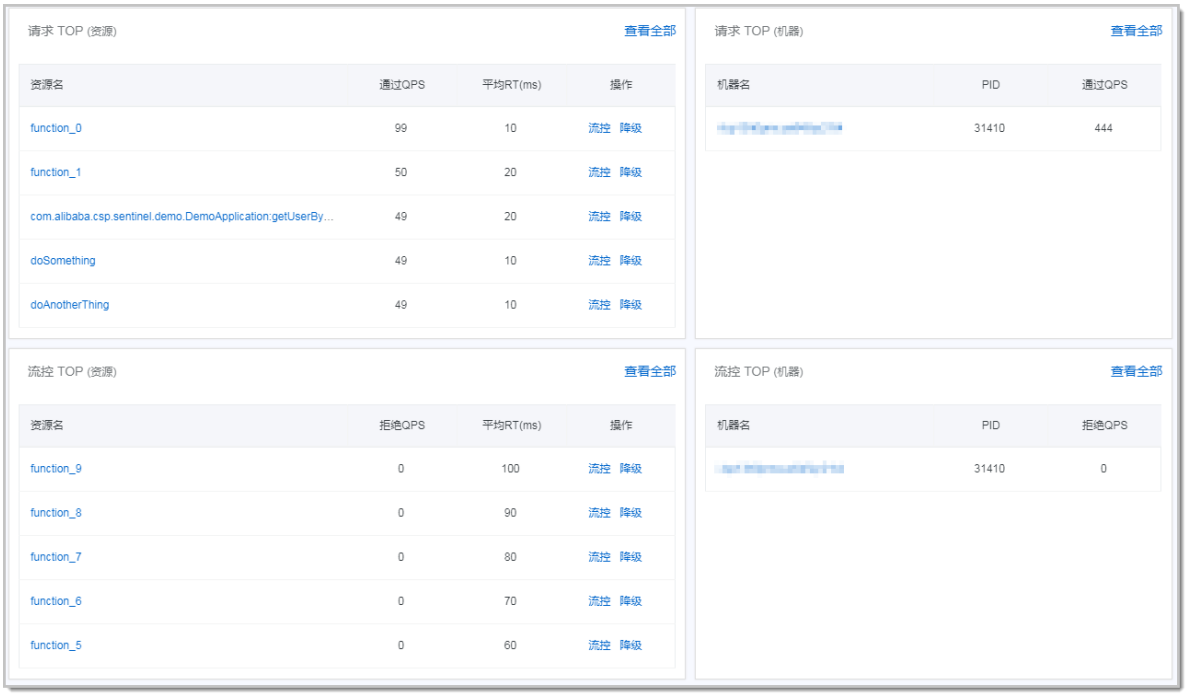
功能介绍

应用概览页面展示了应用的限流指标详情，以 TOP 形式的请求、流控、响应时间、异常等信息。

- 限流指标详情包括：近 5 分钟的通过请求数、限流请求数、异常请求数以及环比、同比升降幅度；该应用配置的规则数；集群通过 QPS、集群限流 QPS、集群异常 QPS、集群平均响应时间以及它们的时序曲线图。

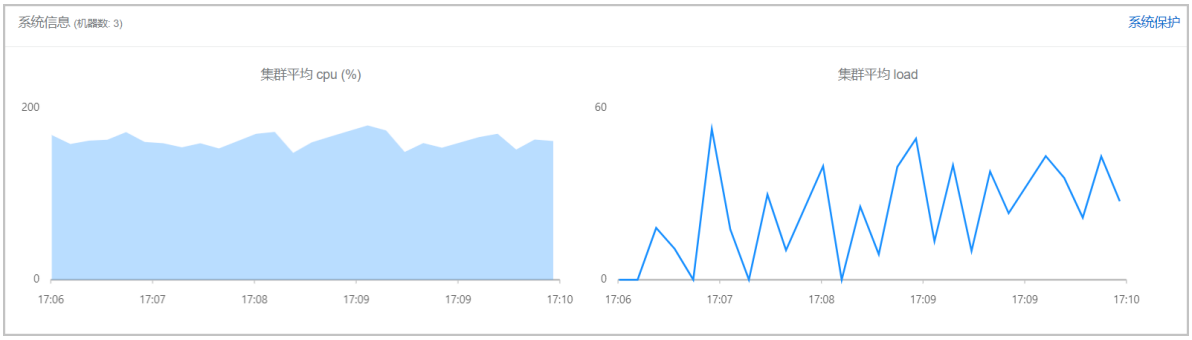
- 单击**规则数**区域的规则类型可以查看具体规则详情。
- 单击时间图标可以自定义时序曲线的时间区间。

- Top 列表包括请求、流控、响应时间、异常等资源信息，以及它们对应的服务器信息。



- 单击目标资源**操作**列的**流控**或**降级**，可为该资源配置相应规则，参见配置流控规则和配置降级规则。
- 单击资源名称可以查看该资源的监控详情。

- 系统信息包含集群的平均 CPU 和集群平均负载的时序曲线。



- 单击**系统保护**可以为应用创建系统规则，请参见配置系统规则。
- 移动鼠标至曲线图中，可查看具体时间点集群 CPU 或负载详细信息。

3.4.2. 监控详情

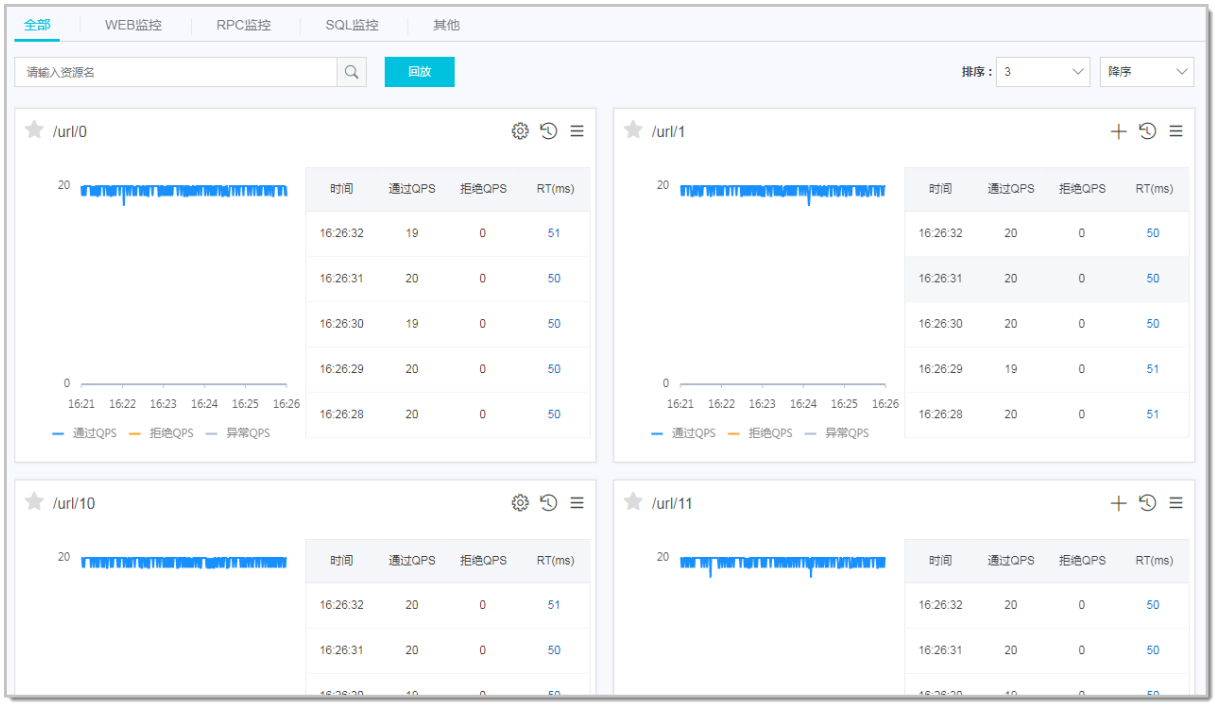
在**监控详情**页面，您可以查看应用的所有资源、以及这些资源的 QPS 时序曲线和时序列表。

功能入口

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**应用防护**。
3. 在**应用列表**页面单击目标应用的资源卡片。
4. 在左侧导航栏中选择**监控详情**。

功能介绍

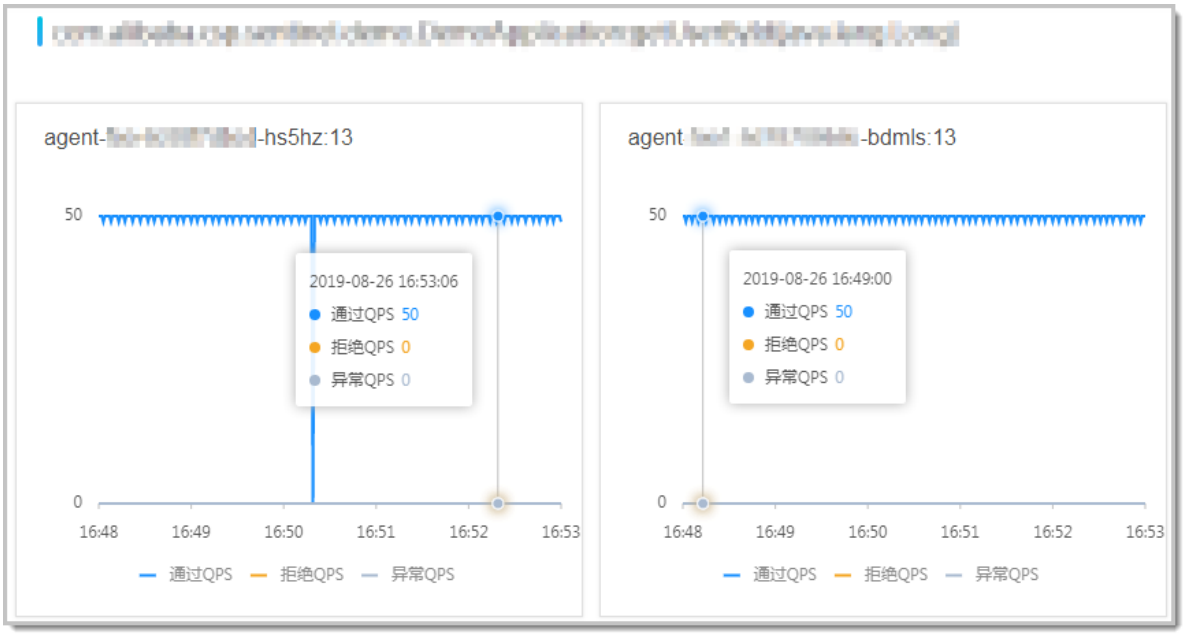
监控详情页面展示了应用的所有资源以及这些资源的 QPS 时序曲线和时序列表。



您可以在**监控详情**页面进行以下操作：

- 在顶部导航栏中单击各类型的页签可以过滤对应类型的资源。

- 单击资源卡片右上角的加号图标，可以为该资源新增流控或降级规则，请参见配置流控规则和配置降级规则。
- 单击资源卡片上的时间图标可以自定义时序曲线的时间区间。
- 单击资源卡片右上角的设置图标，可以查看该资源在所有服务器上的 QPS 详情。



- 单击时序列表的具体 RT 数值，可以查看 RT 的时序分布曲线。

3.4.3. 簇点链路

在簇点链路页面，您可以查看单个应用在单台服务器（实例）上的所有资源以及实时的调用数据。

功能入口

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**应用防护**。
3. 在**应用列表**页面单击目标应用的资源卡片。
4. 在左侧导航栏中选择**簇点链路**。

功能介绍

簇点链路页面展示了单个应用在单台服务器（实例）上的所有资源以及实时的调用数据。

> **说明** 只有当应用中的资源有访问量时，才会显示在**簇点链路**页面。

簇点链路

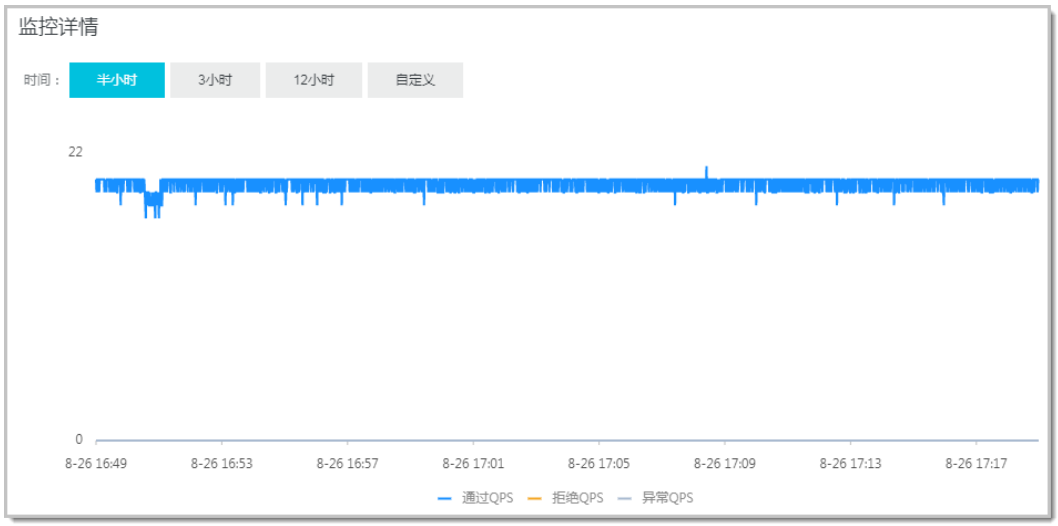
您可以在**簇点链路**页面进行以下操作：

- 在**机器**右侧下拉框中选择目标服务器 IP 地址，可快捷切换服务器。
- 在页面右侧单击**平铺展示**和**树状展示**，可快速变更展示方式。
  - 平铺视图：不区分调用链路关系，平铺展示资源的运行情况。
  - 树状视图：根据资源的调用链路关系，展示树状结构。
- 单击目标资源**操作**列中的**流控**或**降级**，可以快速创建流控和降级规则。具体操作，请参见配置流控规则和配置降级规则。

  > **说明** 在请求链路**簇点链路**页面新建的流控或降级规则，在全局生效。

- 单击目标资源**操作**列中的**监控**，可查看自定义时间段内通过的 QPS 、被拦截的 QPS 和异常 QPS 的时序曲线。

  监控详情



4.网关防护

4.1. 接入应用

4.1.1. 接入 Spring Cloud Gateway 应用

Spring Cloud Gateway 应用可以通过 SDK 接入的方式接入 AHAS 网关防护。将 Spring Cloud Gateway 应用接入 AHAS 网关防护后，可以对其配置流控规则来保证系统稳定性。本文介绍如何使用 SDK 方式将 Spring Cloud Gateway 应用接入网关防护。

操作步骤

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**网关防护**。
3. 在**网关列表**页面右上角单击**新应用接入**，然后单击 **Spring Cloud Gateway 网关接入**页签。
4. 在 Spring Cloud Gateway 应用的 Pom 文件中添加以下依赖：

```
<dependency>
  <groupId>com.alibaba.csp</groupId>
  <artifactId>spring-cloud-gateway-starter-ahas-sentinel</artifactId>
  <version>x.y.z</version>
</dependency>
```

> **说明** 在 Spring Cloud Gateway 网关接入页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

5. 配置应用的启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

6. 重启网关应用。

结果验证

完成上述步骤后，在**Spring Cloud Gateway 网关接入**页面单击**我已完成上述步骤**。若该网关有访问量，则在 AHAS 控制台的**网关列表**页面看到该网关服务。

后续步骤

接入网关应用后，可以为该应用配置网关流控规则。

- 配置网关流控规则
- API 管理

4.1.2. 接入 Spring Cloud Zuul 应用

Spring Cloud Zuul 应用可以通过 SDK 接入的方式接入 AHAS 网关防护。将 Spring Cloud Zuul 应用接入 AHAS 网关防护后，可以对其配置流控规则来保证系统稳定性。本文介绍如何使用 SDK 方式将 Spring Cloud Zuul 应用接入网关防护。

操作步骤

1. 登录 AHAS 控制台。
2. 在左侧导航栏，单击**网关防护**。
3. 在**网关列表**页面右上角单击**网关接入**，然后单击**Zuul（1.x） 网关接入**页签。
4. 在 Spring Cloud Zuul 应用的 Pom 文件中添加以下依赖：

```
<dependency>
  <groupId>com.alibaba.csp</groupId>
  <artifactId>spring-cloud-zuul-starter-ahas-sentinel</artifactId>
  <version>x.y.z</version>
</dependency>
```

> **说明** 在 Zuul（1.x） 网关接入页签查看 Pom 依赖最新版本，将 `x.y.z` 替换为新版本的版本号。

5. 通过以下任意一种方式，配置应用的启动参数。

```
-Dahas.namespace=default
-Dahas.gateway.address=<Address>
-Daddress.server.domain=<Admain>
-Dproject.name=<AppName>
```

> **说明** 将 <AppName> 替换为应用名称，<Address> 替换为环境地址，<Domain> 替换为环境域名。

6. 重启网关应用。

结果验证

完成上述步骤后，在**Zuul（1.x） 网关接入**页面单击**我已完成上述步骤**。若该网关有访问量，则在 AHAS 控制台的**网关列表**页面看到该网关服务。

后续步骤

接入网关应用后，可以为该应用配置网关流控规则。

- 配置网关流控规则
- API 管理

4.2. 控制台操作

4.2.1. 配置网关流控规则

为网关应用配置网关流控规则后，AHAS 将从流量入口处拦截激增的流量，防止下游服务被压垮。本文将介绍如何为已接入 AHAS 的网关应用配置网关流控规则。

前提条件

应用已接入 AHAS，详情请参见：

- 接入 Spring Cloud Gateway 应用
- 接入 Spring Cloud Zuul 应用

新建网关流控规则

1. 登录 AHAS 控制台。
2. 在左侧导航栏，单击**网关防护**。
3. 单击目标网关应用卡片，然后任选一种方式进入网关流控规则的配置页面：
   - 在**监控详情**页面，单击 API 资源卡片右上角的加号图标。
   - 在左侧导航栏中单击**流控规则**。然后在**流控规则**页面右上角单击**新增流控规则**。
4. 在**新增流控规则**对话框中，配置流控规则。

   - **API**：选择适用该规则的自定义 API，或者手动输入路由配置文件中的 Route ID。
   - **针对请求属性**：可选择闭合或打开。

- 闭合**针对请求属性**开关：不针对请求属性（如 Client IP，URL 参数等）进行限流，直接针对该 API 的所有请求进行流量控制。
- 开启**针对请求属性**开关：针对该 API 的某个请求属性进行限流。例如，可以针对每个 URL 参数的访问进行限制。

- **阈值类型**
  - **QPS**：应用或服务流量的 QPS 指标。选择 QPS 后，还需设置 QPS 阈值和统计间隔（支持秒、分钟、小时、天 4 种维度）。

    例如，QPS 阈值填写 10，统计间隔选择**分**，则代表每分钟对应的请求数目不超过 10 个。
  - **线程数**：资源的并发线程数，即该资源正在执行的线程数。

    > **说明** 开启**针对请求属性**开关后，暂时不支持线程数作为阈值类型。

- **参数属性**
  - **Client IP**：请求端的 IP 地址。
  - **Remote Host**：请求端的 Host Header。
  - **Header**：根据指定的 HTTP Header 进行解析，匹配对应的 Header Key。选择 **Header** 后，可以配置请求属性值的匹配策略，只有匹配该模式的请求属性值会纳入统计和流控。
    - **精确**：严格按照给定的匹配串来匹配值。
    - **子串**：若请求属性值包含该子串则匹配成功，如子串匹配 `ab`，则 `aba` 和 `cabc` 都可以匹配，而 `cba` 则不能匹配。
    - **正则**：按照给定的正则表达式匹配串来进行匹配。
  - **URL 参数**：根据指定的 HTTP URL 参数进行解析，需要填写对应的参数名称。选择 **URL 参数**后，可以配置请求属性值的匹配策略，只有匹配该模式的请求属性值会纳入统计和流控。

- **流控方式**
  - **快速失败**：当**阈值类型**为 **QPS** 时，被拦截的流量将快速失败。即达到阈值时，立即拦截请求。
  - **匀速排队**：当**阈值类型**为 **QPS** 时，被拦截的请求将匀速通过，允许排队等待。

    需设置具体的超时时间，预计达到超时时间的请求会立即失败，而不会排队。

    例如，QPS 配置为 10，则代表请求每 100 ms 才能通过一个，多出的请求将排队等待通过。超时时间代表最大排队时间，超出最大排队时间的请求将会直接被拒绝。

    > **说明** 匀速排队时，QPS 不要超过 1000（请求间隔 1 ms）。

  - **Burst size**：当流控方式为**快速失败**时，可以额外设置一个 Burst Size，即针对突发请求额外允许的请求数目。
  - **超时时间**：当流控方式为**匀速排队**时，需设置具体的超时时间，达到超时时间后请求会失败。例如，QPS 配置为 5，则代表请求每 200 ms 才能通过一个，多出的请求将排队等待通过。超时时间代表最大排队时间，超出最大排队时间的请求将会直接被拒绝。

5. 单击**新增**。

   新增的规则将出现在**流控规则**页面。

管理流控规则

在**流控规则**页面，您可以启用、禁用、编辑或删除流控规则。

- 单流控规则启用/禁用：

  在**流控规则**页面，找到目标资源下对应的流控规则，单击状态**状态**栏的启用开关，可快速启用或禁用该规则。

- 多流控规则批量启用/禁用：

  在**流控规则**页面，勾选多个流控规则，单击**批量启用**或**批量禁用**，可快速启用或禁用多个规则。

- 编辑规则：

  在**流控规则**页面，找到目标资源下对应的流控规则，单击**操作**栏的**编辑**，可修改该规则的相关信息。

- 删除规则：

  在**流控规则**页面，找到目标资源下对应的流控规则，单击**操作**栏的删除**删除**。

4.2.2. API 管理

在 AHAS 网关防护中，您可以创建 API 分组，并自定义每个 API 下面的 URL 路径匹配规则。AHAS 网关防护可以针对自定义的 API 分组进行流量控制。

新建自定义 API

按照以下步骤新建需要流控的 API 分组。

1. 登录 AHAS 控制台。
2. 在左侧导航栏中选择**网关防护**。
3. 单击目标网关应用卡片，进入该网关的监控详情。
4. 在左侧导航栏，选择 **API 管理**。单击右上角的**新增**按钮。
5. 在**新建自定义 API** 对话框中，填写 API 分组名称。该名称需要全局唯一，并且不能与路由配置文件中的路由 ID 重复。

6. 填写 URL 路径匹配规则，先选择匹配模式，再根据匹配模式的要求填写匹配串。
   - **匹配模式**分为以下三类：
     - **精确模式**：严格按照给定的匹配串来匹配 URL 路径。示例：`/foo` 代表严格按照 `/foo` 这个路径来匹配。
     - **前缀模式**：按照给定的匹配串来进行前缀匹配，匹配串需符合 Spring Web 风格。示例：`/foo/**` 代表匹配以 `/foo/` 开头的所有 URL，像 `/foo/22` 这种 URL 都可以匹配。

- **正则模式**：按照给定的正则表达式匹配串来进行匹配。

- **匹配串**：根据匹配模式的要求填写匹配串。

7. 单击**+新增匹配规则**，可添加多个 URL 路径匹配规则。
8. 单击**新增**，完成自定义 API 的创建。

   新增的 API 将出现在 **API 管理**页面。

后续操作

新增 API 后，您可以编辑、删除 API，并对自定义的 API 设置流控规则。

- 编辑 API
  i. 在 **API管理**页面，在目标 API 的**操作**列，单击**编辑**。
  ii. 在**编辑自定义API** 对话框中，修改 URL 匹配规则，也可以新增 URL 匹配规则。
- 删除 API
  i. 在 **API管理**页面，在目标 API 的**操作**列，单击**删除**。
  ii. 在提示框中，单击**确定**，将该 API 分组删除。
- 设置流控规则

  请参见配置网关流控规则。

5.参考信息

5.1. 支持组件列表

AHAS为了简化应用的接入流程，对主流框架进行了适配。本文列出了AHAS支持的第三方组件和框架列表。

组件	支持版本	支持该组件的Java Agent版本	支持该组件的SDK版本
Dubbo	Agent：2.6.x SDK：2.7.x，2.6.x	All	All
Web Servlet	SDK：2.x+ Agent：3.0+	All	All
Spring Boot	1.3.x+	All	请参见说明
Spring MVC	4.x+	All	请参见说明
Spring Cloud Gateway	2.x	1.5.0+	请参见说明
Zuul 1.x	1.3.x	1.5.0+	请参见说明
GRPC-Java	1.13+	1.7.0	需另引入sentinel-grpc-adapter依赖
Jetty	8.x+	Servlet 3.0+ 支持：all	通过Servlet支持
Tomcat	7.x+	Servlet 3.0+ 支持：all	通过Servlet支持
WebLogic	10.3	Servlet 3.0+ 支持：all Servlet 2.x支持：1.6.0+	通过Servlet支持
HttpClient 3	3.x+	待支持	待支持
HttpClient 4	4.x+	待支持	待支持
JDK HTTP	1.7.x+	待支持	待支持
OKHttp	2.x+	待支持	待支持
MyBatis	3.x+	待支持	1.4.1+
MySQL JDBC	5.0.x+	1.6.0+	不支持
Oracle JDBC	12.x	1.6.0+	不支持
PostgreSql JDBC	9.4+	待支持	不支持

组件	支持版本	支持该组件的Java Agent版本	支持该组件的SDK版本
SQLServer JDBC	6.4+	待支持	不支持
Redis Client（Jedis）	待支持	1.7.0	待支持
MemCached	2.8+	1.7.0	待支持
MongoDB	3.7+	待支持	待支持
RocketMQ （callback模式）	4.x	1.7.0	需手动埋点
RabbitMQ	3.7+	1.7.0	需手动埋点
SOFARPC	5.x	待支持	1.5.3+

说明

- Spring Boot/Spring Cloud Web应用只需要引入 `spring-boot-starter-ahas-sentinel-client` 依赖即可接入。
- Spring Cloud Gateway网关需要引入 `spring-cloud-gateway-starter-ahas-sentinel` 依赖；Zuul 1.x网关需要引入 `spring-cloud-zuul-starter-ahas-sentinel` 依赖，无需引入其它依赖。

5.2. 性能基准

本文列出了 AHAS 应用流控降级在特定 CPU、OS、Java 版本的测试环境下的基准表现。

测试环境

基准的测试环境：

- CPU：Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz （32 cores）
- OS：Linux 2.6.32-220.23.2.ali927.el5.x86_64
- Java 版本：

```
java version "1.8.0_77"
Java(TM) SE Runtime Environment (build 1.8.0_77-b03)
Java HotSpot(TM) 64-Bit Server VM (build 25.77-b03, mixed mode)
```

吞吐量对比

所有吞吐量测试都基于 JMH 编写。

- 单线程吞吐量：

  JVM 参数： `-Xmn256m -Xmx1024m -XX:+UseConcMarkSweepGC`

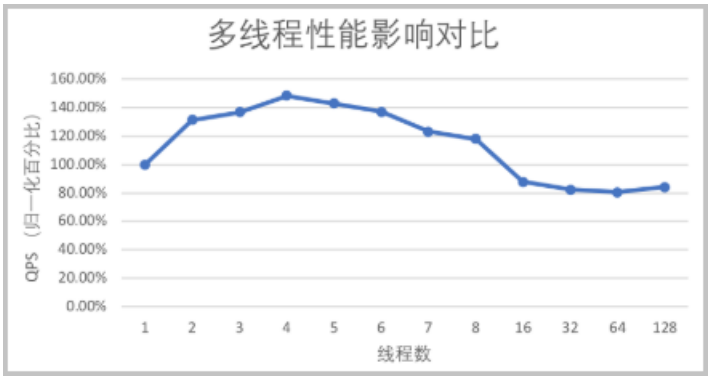
  通过执行 CPU 密集型操作（如小数组排序）模拟不同 QPS 下的情况，来测试单线程模式下接入 AHAS 应用流控降级与不接入 AHAS 应用流控降级的对比。测试逻辑见 SentinelEntryBenchmark。相关结果如下：

数组长度	Baseline（QPS）	With Sentinel（QPS）	性能损耗
length=25	604589.916	401687.765	33.56%
length=50	221307.617	192407.832	13.06%
length=100	97673.228	91535.146	6.28%
length=200	43742.960	42711.129	2.36%
length=500	15332.924	15171.024	1.06%
length=1000	7012.848	6984.036	0.41%

  - 当单机 QPS 为 20 万以上时，AHAS 应用流控降级带来的性能损耗比较大。这种情况业务场景（如缓存读取操作）本身的耗时非常小，而 AHAS 应用流控降级的统计、检查操作会消耗一定的时间。
  - 单机 QPS 在 5 万以下时，AHAS 应用流控降级性带来的能损耗比较小，适用于大多数场景。

- 多线程吞吐量影响：

  在相同逻辑（对 length=25 的数组进行 shuffle 并排序）的情况下，测试不同线程数下接入 AHAS 应用流控降级的性能表现：



内存占用情况

测试场景：6000 个资源循环执行（即单机的极端场景，目前最多支持 6000 个 Entry）。

- 单线程循环执行：内存占用约 185 MB。
- 8 线程循环执行：内存占用约 1 GB（若系统持续高并发持续很，将导致底层的 LongAdder 内存占用很高）。

5.3. 客户端启动参数

将应用接入 AHAS 应用防护时，您可以在启动命令中修改客户端启动参数来控制客户端行为。本文列出了 AHAS 支持的客户端启动参数。

名称	含义	类型	默认值	备注
-Dproject.name	指定应用的名称，会显示在 AHAS 应用列表中	String	Main 函数所在的类全名	非必需，建议配置

名称	含义	类型	默认值	备注
-Dcsp.sentinel.app.type	指定应用的类型	int	0（APP_TYPE_COMMON）	非必需，1.3.0+版本支持，一般不需要手动配置
-Dcsp.sentinel.metric.file.single.size	单个监控日志文件的大小	long	52428800（50MB）	非必需
-Dcsp.sentinel.metric.file.total.count	监控日志文件的总数上限	int	6	非必需
-Dcsp.sentinel.statistic.max.rt	最大的有效响应时长（ms），超出此值则按照此值记录	int	4900	非必需
-Dcsp.sentinel.log.dir	Sentinel 日志文件目录	String	${user.home}/logs/csp/	非必需
-Dcsp.sentinel.log.use.pid	日志文件名中是否加入进程号，用于单机部署多个应用的情况	boolean	false	非必需
-Dcsp.sentinel.log.output.type	Record 日志输出的类型，`file` 代表输出至文件，`console` 代表输出至终端	String	file	非必需，1.3.6+版本支持
-Dcsp.sentinel.heartbeat.interval.ms	心跳包发送周期，单位毫秒	long	5s	非必需，若不进行配置，则会从相应的 `HeartbeatSender` 中提取默认值
-Dcsp.sentinel.web.servlet.block.page	限流页	String	null	非必需
-Dcsp.sentinel.url.clean.config.path	URL 收敛规则配置文件路径	String	空	非必需
-Dcsp.sentinel.url.suffix.exclude.pattern	收敛的URL后缀	String	png,gif,mjs,css,htm,html,jpg,jpeg,map,ico,ttf,woff	非必需

> **说明** 列表中的版本信息即 ahas-sentinel-client 的版本。

6.流控降级常见问题

1. **应用列表**或**网关列表**页面看不到应用？

   请参考以下步骤检查：

   - 确保引入了相应的依赖，并进行了正确的配置。
   - 确保应用资源有访问量。
   - 确保本机时间正确：程序所在的机器时间要跟当地标准时间一致。

2. **簇点链路**页面没有看到想要的资源？

   请确保对应资源有访问量。资源要有访问量才会在请求链路页面显示。另外请求链路页面显示的是单台机器上的所有调用链路数据，您可以通过下拉框切换机器查看不同机器的簇点链路数据。

3. **簇点链路**页面添加规则是给单台机器添加还是给所有机器都添加？

   添加规则会推送到所有机器。

4. 流控规则中的来源应用是什么意思？

   Sentinel 支持按调用来源限流。流控规则中来源应用（针对应用）指的是调用该资源的调用方标识，例如在 Dubbo 中就对应 Dubbo Consumer 的应用名称。默认来源应用设为 `default`，代表不区分来源应用。

5. 流控规则中的线程数模式是什么意思？

   线程数模式按照资源的**并发线程数**（即该资源正在执行的线程数）进行流量控制。

6. 流控规则中的每种流控模式是什么意思？

   - 直接：直接按照当前资源的**调用来源**进行限流，若来源为 `default` 则不区分调用来源
   - 关联：当两个资源之间具有资源争抢或者依赖关系的时候，这两个资源便具有了关联。关联限流会根据当前资源的**关联资源**进行限流。

     比如 read_db 和 write_db 这两个资源分别代表数据库读写，我们可以给 read_db 设置限流规则来达到写优先的目的：设置 **关联资源** 为 write_db。这样当写库操作过于频繁时，读数据的请求会被限流。
   - 链路：根据调用链路入口限流。需要在规则中配置**入口资源**，即该调用链路入口的上下文名称。

7. 流控规则中的每种流控方式是什么意思？

   - 快速失败：达到阈值时，立即拦截请求。
   - Warm Up：当流量突然增大的时候，我们希望系统从空闲状态到繁忙状态的切换的时间长一些，即如果系统在此之前长期处于空闲的状态，我们希望处理请求的速率缓慢增加，经过预期的时间以后，到达系统处理请求速率的设定值；默认会从配置 QPS 阈值的 1/3 开始慢慢往上增加 QPS。
   - 排队等待：请求匀速通过，允许排队等待，通常用于消息队列削峰填谷等场景。比如配置了 QPS 为 5，则代表请求每 200 ms 才能通过一个，多出的请求将排队等待通过。超时时间代表最大排队时间，超出最大排队时间的请求将会直接被拒绝。**注意排队等待模式下 QPS 不要超过 1000（请求间隔 1 ms）。**

8. 降级规则中的 RT 模式/异常比例模式是什么意思？

   Sentinel 熔断降级支持两种策略：

   - RT：按照**秒级的平均响应时间**进行降级，单位是毫秒。如果持续进入 5 个请求，对应的平均 RT 都持续超过降级阈值，则会被自动降级。

- 异常比例：按照**秒级的异常比例**进行降级。当资源的每秒异常总数占通过量的比值超过阈值之后，资源进入降级状态。

资源进入降级状态后，在配置的降级窗口时间内，请求都会快速失败。

9. 什么是系统保护规则？

   系统保护规则可以从系统指标维度（入口 QPS、RT、线程数、Load 等）来进行流量控制，让系统尽可能跑在最大吞吐量水位的同时保证系统整体的稳定性。系统保护规则是整体维度的而不是资源维度的，并且仅对入口流量生效。入口流量指的是进入应用的流量，比如 Web 服务或 Dubbo 服务端接收的请求，都属于入口流量。

   系统规则支持的种类有以下几种：

   - **系统 Load**（load1）：按照系统 load1 以及实时的吞吐量进行流量控制。
   - **总体平均 RT**：当单台机器上所有入口流量的平均 RT 达到阈值即触发系统保护。
   - **总体 QPS**：当单台机器上所有入口流量的 QPS 达到阈值即触发系统保护。
   - **总体线程数**：当单台机器上所有入口流量的并发线程数达到阈值即触发系统保护。

   对于一个应用来说，每种相同的系统保护规则最多只能存在一条。