Hystrix Dashboard: 断路器执行监控

Hystrix Dashboard 是Spring Cloud中查看Hystrix实例执行情况的一种仪表盘组件,支持查看单个实例和查看集群实例,本文将对其用法进行详细介绍。

简介

Hystrix提供了Hystrix Dashboard来实时监控HystrixCommand方法的执行情况。 Hystrix Dashboard可以有效地反映出每个Hystrix实例的运行情况,帮助我们快速发现系统中的问题,从而采取对应措施。

Hystrix 单个实例监控

我们先通过使用Hystrix Dashboard监控单个Hystrix实例来了解下它的使用方法。

创建一个hystrix-dashboard模块

用来监控hystrix实例的执行情况。

• 在pom.xml中添加相关依赖:

```
<dependency>
2
         <groupId>org.springframework.cloud</groupId>
3
         <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
4
     </dependency>
5
     <dependency>
6
         <groupId>org.springframework.cloud
 7
         <artifactId>spring-cloud-starter-netflix-hystrix-dashboard</artifactId>
8
     </dependency>
9
    <dependency>
10
         <groupId>org.springframework.boot</groupId>
11
         <artifactId>spring-boot-starter-actuator</artifactId>
     </dependency>Copy to clipboardErrorCopied
12
```

• 在application.yml进行配置:

```
server:
2
       port: 8501
3
     spring:
4
       application:
5
         name: hystrix-dashboard
   eureka:
6
7
       client:
8
         register-with-eureka: true
9
         fetch-registry: true
10
         service-url:
           defaultZone: http://localhost:8001/eureka/Copy to clipboardErrorCopied
11
```

• 在启动类上添加@EnableHystrixDashboard来启用监控功能:

```
1  @EnableHystrixDashboard
2  @EnableDiscoveryClient
3  @SpringBootApplication
4  public class HystrixDashboardApplication {
5     public static void main(String[] args) {
6         SpringApplication.run(HystrixDashboardApplication.class, args);
7         SpringApplication.run(HystrixDashboardApplication.class, args);
8     }
9 }
10 }Copy to clipboardErrorCopied
```

启动相关服务

这次我们需要启动如下服务: eureka-server、user-service、hystrix-service、hystrix-dashboard,启动后注册中心显示如下。

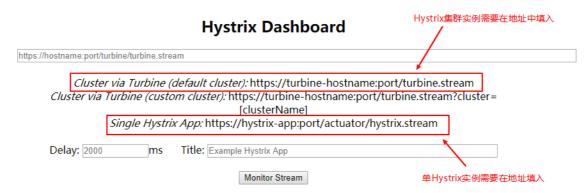
Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
HYSTRIX-DASHBOARD	n/a (1)	(1)	UP (1) - 192.168.56.1:hystrix-dashboard:8501
HYSTRIX-SERVICE	n/a (1)	(1)	UP (1) - 192.168.56.1:hystrix-service:8401
USER-SERVICE	n/a (1)	(1)	UP (1) - 192.168.56.1:user-service:8201

Hystrix实例监控演示

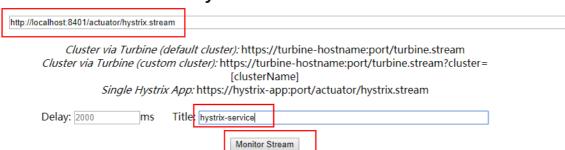
• 访问Hystrix Dashboard: http://localhost:8501/hystrix





• 填写好信息后点击监控按钮,这里我们需要注意的是,由于我们本地不支持https,所以我们的地址需要填入的是http,否则会无法获取监控信息;

Hystrix Dashboard



• 还有一点值得注意的是,被监控的hystrix-service服务需要开启Actuator的hystrix.stream端点,配置信息如下:

```
1 management:
2 endpoints:
3 web:
4 exposure:
5 include: 'hystrix.stream' #暴露hystrix监控端点Copy to clipboardErrorCopied
```

• 调用几次hystrix-service的接口: http://localhost:8401/user/testCommand/1

Hystrix Stream: hystrix-service



Circuit Sort: Error then Volume | Alphabetical | Volume | Error | Mean | Median | 90 | 99 | 99.5

Success | Short-Circuited | Bad Request | Timeout | Rejected | Failure | Error %

getUserCommand

| Timeout | Rejected | Failure | Error %

| GetUser: 0.7/s | Circuit Closed |

• 可以发现曾经我们在@HystrixCommand中添加的commandKey和threadPoolKey属性都显示在上面了,并且有7次调用都成功了。

Hystrix Dashboard 图表解读

图表解读如下,需要注意的是,小球代表该实例健康状态及流量情况,颜色越显眼,表示实例越不健康,小球越大,表示实例流量越大。曲线表示Hystrix实例的实时流量变化。

Hystrix Stream: hystrix-service





Hystrix 集群实例监控

这里我们使用Turbine来聚合hystrix-service服务的监控信息,然后我们的hystrix-dashboard服务就可以从Turbine获取聚合好的监控信息展示给我们了。

创建一个turbine-service模块

用来聚合hystrix-service的监控信息。

• 在pom.xml中添加相关依赖:

```
1
     <dependency>
 2
         <groupId>org.springframework.cloud
 3
         <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
 4
     </dependency>
 5
     <dependency>
         <groupId>org.springframework.cloud
6
 7
         <artifactId>spring-cloud-starter-netflix-turbine</artifactId>
     </dependency>
8
9
     <dependency>
10
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-actuator</artifactId>
11
     </dependency>Copy to clipboardErrorCopied
12
```

• 在application.yml进行配置,主要是添加了Turbine相关配置:

```
server:
2
       port: 8601
3
     spring:
4
       application:
 5
         name: turbine-service
 6
     eureka:
       client:
 8
          register-with-eureka: true
9
         fetch-registry: true
10
         service-url:
```

```
11 defaultZone: http://localhost:8001/eureka/
12 turbine:
13 app-config: hystrix-service #指定需要收集信息的服务名称
14 cluster-name-expression: new String('default') #指定服务所属集群
15 combine-host-port: true #以主机名和端口号来区分服务Copy to clipboardErrorCopied
```

• 在启动类上添加@EnableTurbine来启用Turbine相关功能:

```
@EnableTurbine
1
2
     @EnableDiscoveryClient
3
     @SpringBootApplication
     public class TurbineServiceApplication {
4
6
         public static void main(String[] args) {
7
             SpringApplication.run(TurbineServiceApplication.class, args);
8
         }
9
   }Copy to clipboardErrorCopied
10
```

启动相关服务

使用application-replica1.yml配置再启动一个hystrix-service服务,启动turbine-service服务,此时注册中心显示如下。

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
HYSTRIX-DASHBOARD	n/a (1)	(1)	UP (1) - 192.168.56.1:hystrix-dashboard:8501
HYSTRIX-SERVICE	n/a (2)	(2)	UP (2) - 192.168.56.1:hystrix-service:8402 , 192.168.56.1:hystrix-service:8401
TURBINE-SERVICE	n/a (1)	(1)	UP (1) - 192.168.56.1:turbine-service:8601
USER-SERVICE	n/a (1)	(1)	UP (1) - 192.168.56.1:user-service:8201

Hystrix集群监控演示

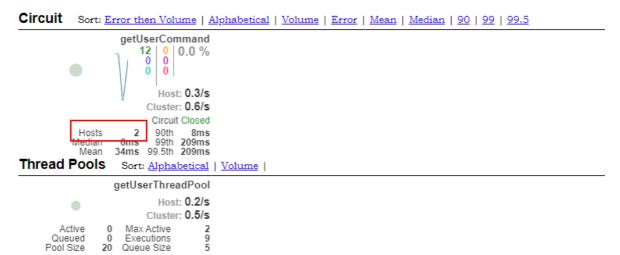
- 访问Hystrix Dashboard: http://localhost:8501/hystrix
- 添加集群监控地址,需要注意的是我们需要添加的是turbine-service的监控端点地址:

Hystrix Dashboard

/localhost:8601/turbine.stream	
Cluster via Turbine (d	lefault cluster): https://turbine-hostname:port/turbine.stream
Cluster via Turbine (custo	<i>m cluster):</i> https://turbine-hostname:port/turbine.stream?cluster=
	[clusterName]
Single Hystrix	App: https://hystrix-app:port/actuator/hystrix.stream
Dolova 2000 ms	Title
Delay: 2000 ms	Title: hystrix-service
	Monitor Stream

• 调用几次hystrix-service的接口: http://localhost:8401/user/testCommand/1以及http://localhost:8402/user/testCommand/1

Hystrix Stream: hystrix-service



• 可以看到我们的Hystrix实例数量变成了两个。

使用到的模块

```
1 springcloud-learning
2 ├── eureka-server -- eureka注册中心
3 ├── user-service -- 提供User对象CRUD接口的服务
4 ├── hystrix-service -- hystrix服务调用测试服务
5 ├── turbine-service -- 聚合收集hystrix实例监控信息的服务
6 └── hystrix-dashboard -- 展示hystrix实例监控信息的仪表盘
```