1. Hystrix简介

微服务架构往往由很多微服务组成，一个微服务往往直接或者间接的调用很多微服务。当一个微服务down之后，可能会影响很多微服务。

在断路器的帮助下可以实现容错。它是一种将请求包装到外部服务并检测它们何时失败的模式。如果检测到故障，断路器就会打开。所有随后的请求立即返回一个错误，而不是向不健康的服务发出请求。它监视和检测服务是否宕机，并与其他服务发生错误行为。它会拒绝调用，直到再次恢复正常。

2. 在项目中使用Hystrix

⑴在pom.xml中加入hystrix依赖

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-netflix-hystrix</artifactId>

</dependency>

⑵在应用程序主类添加注解@EnableHystrix

⑶在调用其它微服务的接口上创建回退方法。

①controller类的接口请求方法添加注解@HystrixCommand(fallbackMethod = methodName)，并在controller类中定义methodName方法。

②以Feign整合Hystrix为例。

@FeignClient注解中使用fallback=回退类.class，回退类如下(该实现类必须为Bean)：

package com.longmao.service2.proxy;

import com.longmao.service2.dto.Service1Entity;

import org.springframework.stereotype.Component;

@Component

public class Service1ProxyImp implements Service1Proxy {

@Override

public Service1Entity getMinAndMax(){

Service1Entity service1Entity = new Service1Entity();

service1Entity.setMin(0);

service1Entity.setMax(0);

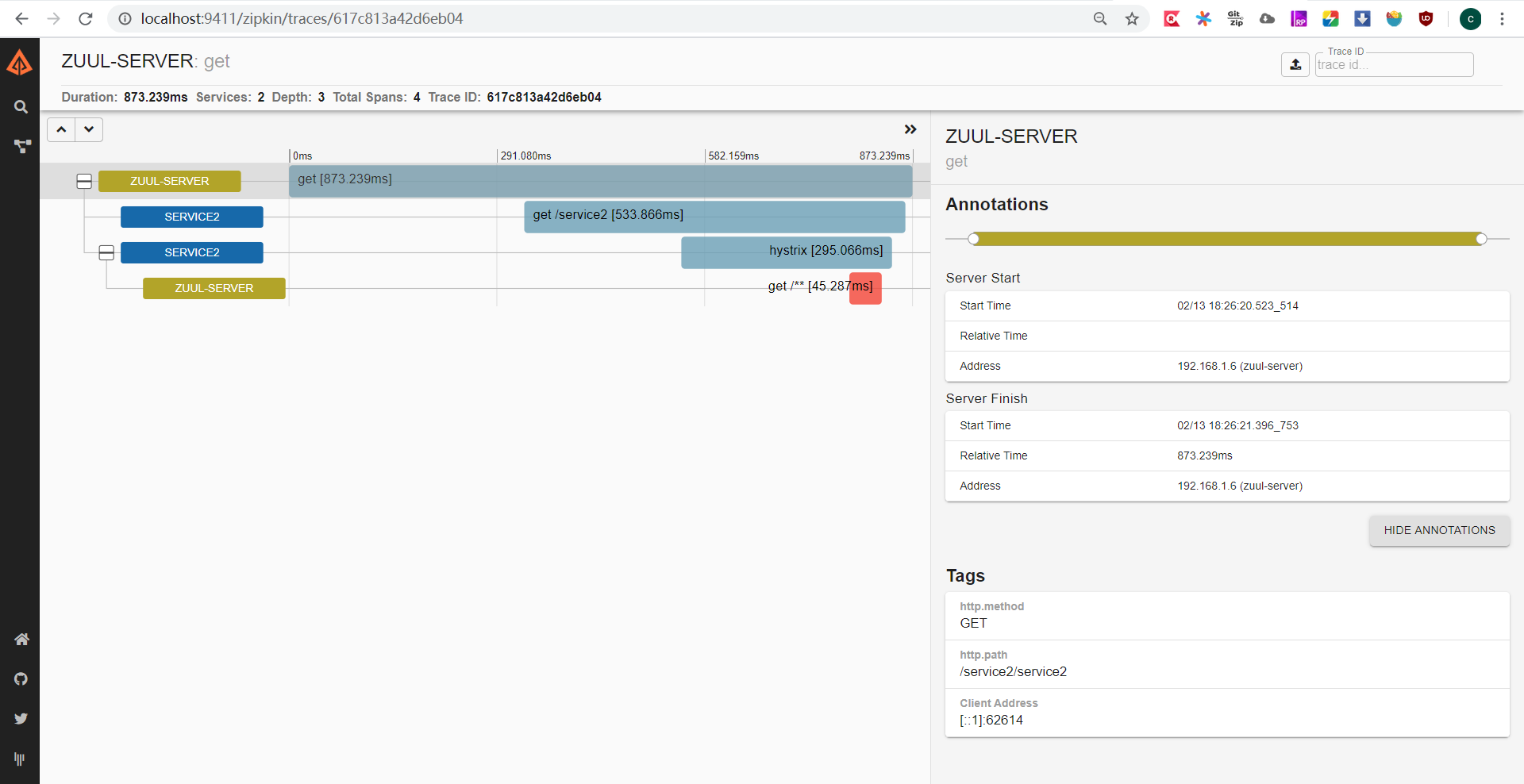
service1Entity.setPort(0);

return service1Entity;

}

}

在Zipkin控制台可以看见接口请求执行详情：

使用fallback可以实现服务不可用时的回退功能，但不知道服务失败的原因。可使用 FallbackFactory 来实现回退功能并可获取触发回退的原因。

在FeignClient中使用fallbackFactory指定回退类，回退类实现FallbackFactory 接口并指定泛型为 @FeignClient 注解的接口。

package com.longmao.service2.proxy;

import com.longmao.service2.dto.Service1Entity;

import feign.hystrix.FallbackFactory;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Component;

@Component

public class Service1ProxyImp implements FallbackFactory<Service1Proxy> {

Logger logger = LoggerFactory.getLogger(this.getClass());

@Override

public Service1Proxy create(Throwable throwable){

return new Service1Proxy() {

@Override

public Service1Entity getMinAndMax(){

Service1Entity service1Entity = new Service1Entity();

service1Entity.setMin(0);

service1Entity.setMax(0);

service1Entity.setPort(0);

logger.info("{}", throwable.getMessage());

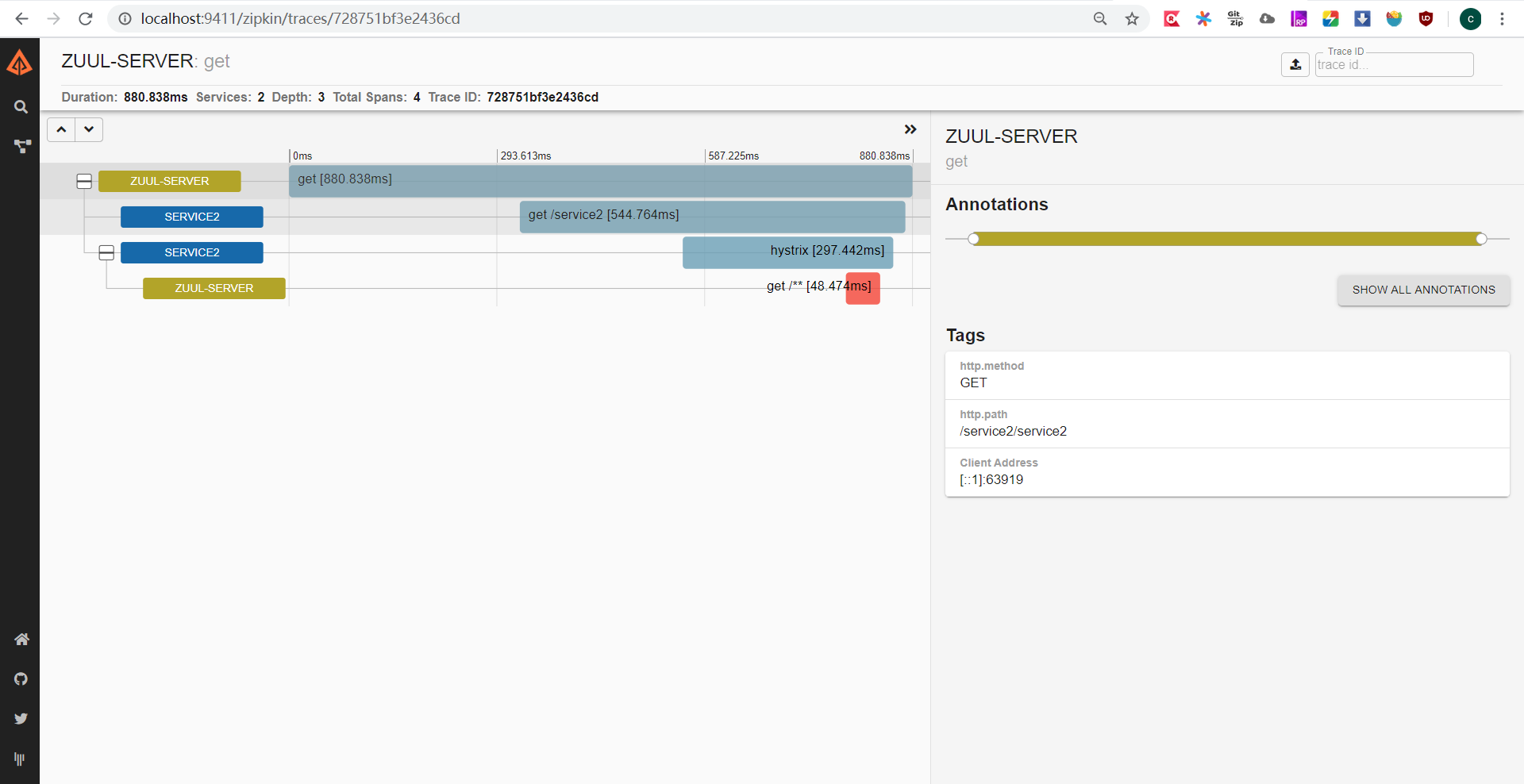
return service1Entity;

}

};

}

}

在Zipkin控制台可以看见接口请求执行详情：