**什么是Sleuth：**

Sleuth是Spring Cloud提供的一个框架，用于追踪微服务的调用过程。当外部用户向集群发起请求时，这些请求将会调用多个微服务，每个微服务又会依赖其他微服务，此时如果出现异常情况，排查问题非常困难，因此我们需要知道到底哪个服务出了问题，哪个环节出了问题，以及Root Cause是什么。Sleuth就是Spring Cloud为解决这些问题精心打造的，它可以很方便的和ZipKin，ELK等数据分析、微服务跟踪系统进行整合。

**Sleuth借鉴了Google Dapper的设计，我们需要了解以下两个概念：**

1）**Trace：表示整个跟踪过程**，从用户发起请求到最终的响应。一次跟踪包含多个跨度，这些跨度一树状结构进行保存。

2）**Span：跨度，表示一次调用的过程**，一次跟踪包含多次调用过程。例如：用户请求A服务，A服务有调用了B服务，那么此时将会产生两个跨度。

除了上面讲的跟踪和跨度外，我们还需要了解以下Annotation（事件标识），它主要用于记录时间的存在，主要包括以下几个事件标识：

cs: Client Sent，客户端发送了请求。

sr: Server Received, 标识服务器接收到了请求，并开始处理。

ss: Server Sent, 标识服务器完成请求的处理，并对客户端做出响应。

cr: Client Received, 标识客户端接收到响应，意味着整个跨度的结束。

**代码演示 (Sleuth搭档Zipkin 实现微服务调用跟踪)：**

项目构成如下：

eureka-server：集群服务器

first-service-provider: 第一个服务提供者

second-service-provider: 第二个服务提供者，为了方便演示跨度

first-service-invoker: 服务调用者

zipkin-server: Zipkin服务器

NOTE：需要源码的盆友请前往github：

https://github.com/aharddreamer/chendong/tree/master/springcloud/sleuth-zipkin-CSDN

**eureka-server依旧只是作为集群服务器，所以不需要改什么代码。**

**zipkin-server我们需要做以下配置：**

首先，POM中确保有这些依赖：

<**dependencyManagement**>  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>org.springframework.cloud</**groupId**>  
 <**artifactId**>spring-cloud-dependencies</**artifactId**>  
 <**version**>Dalston.SR1</**version**>  
 <**type**>pom</**type**>  
 <**scope**>import</**scope**>  
 </**dependency**>  
 </**dependencies**>  
</**dependencyManagement**>

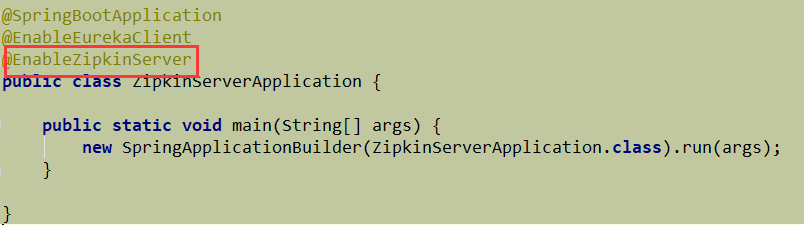
<**dependency**>  
 <**groupId**>org.springframework.cloud</**groupId**>  
 <**artifactId**>spring-cloud-starter-eureka</**artifactId**>  
</**dependency**>

<**dependency**>  
 <**groupId**>io.zipkin.java</**groupId**>  
 <**artifactId**>zipkin-server</**artifactId**>  
</**dependency**>  
  
<**dependency**>  
 <**groupId**>io.zipkin.java</**groupId**>  
 <**artifactId**>zipkin-autoconfigure-ui</**artifactId**>  
</**dependency**>

然后再application.properties或者application.yml文件中加入以下配置：

**spring.application.name**=**zipkin-server  
server.port**=**9411  
eureka.instance.hostname**=**localhost  
eureka.client.service-url.default-zone**=**http://localhost:8761/eureka/**

最后在启动类中加入必要的注解@EnableZipkinServer：



OK了，zipkin服务器就这么搭起来了。

**first-service-provider我们要做这些改动：**

首先确保POM中有如下依赖：

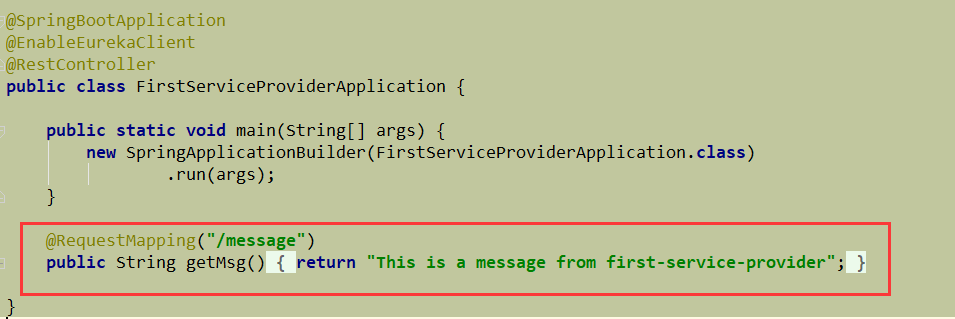
<**dependencies**>  
  
 <**dependency**>  
 <**groupId**>org.springframework.cloud</**groupId**>  
 <**artifactId**>spring-cloud-starter-eureka</**artifactId**>  
 </**dependency**>  
  
 <**dependency**>  
 <**groupId**>org.springframework.cloud</**groupId**>  
 <**artifactId**>spring-cloud-starter-zipkin</**artifactId**>  
 </**dependency**>  
  
 <**dependency**>  
 <**groupId**>org.springframework.cloud</**groupId**>  
 <**artifactId**>spring-cloud-sleuth-zipkin</**artifactId**>  
 </**dependency**>  
  
</**dependencies**>

在application.properties中加入以下配置：

**spring.application.name**=**first-service-provider  
server.port**=**8080  
eureka.instance.hostname**=**localhost  
eureka.client.service-url.default-zone**=**http://localhost:8761/eureka/  
spring.zipkin.base-url**=**http://localhost:9411  
spring.sleuth.sampler.percentage**=**1.0**

最后两行是配置zipkin和sleuth，一个是zipkin服务器的URL，一个是设置追踪的百分比为1.0 也就是100%

我们添加一个测试接口：/message 一会测试用。



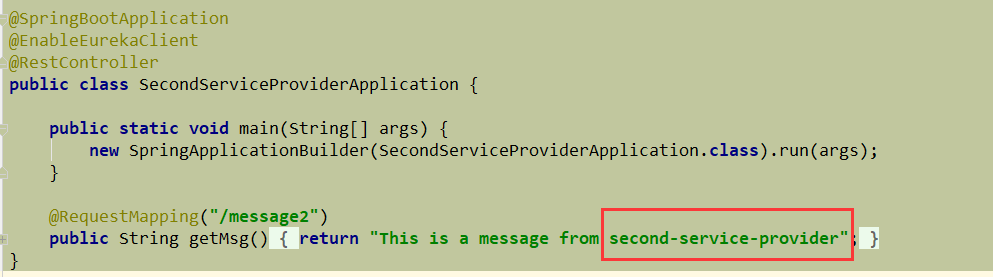
**second-service-provider 中的代码改动：**

POM和first-service-provider中的一样。

application.properties配置如下 （端口和app name不一样）：

**spring.application.name**=**second-service-provider  
server.port**=**8081  
eureka.instance.hostname**=**localhost  
eureka.client.service-url.default-zone**=**http://localhost:8761/eureka/  
spring..base-url**=**http://localhost:9411  
spring.sleuth.sampler.percentage**=**1.0**

也加一个测试接口：/message2



**first-service-invoker 中的改动：**

这个是用户进来的入口，也是调用上面两个service provider的客户端。

POM的依赖和service provider一样。

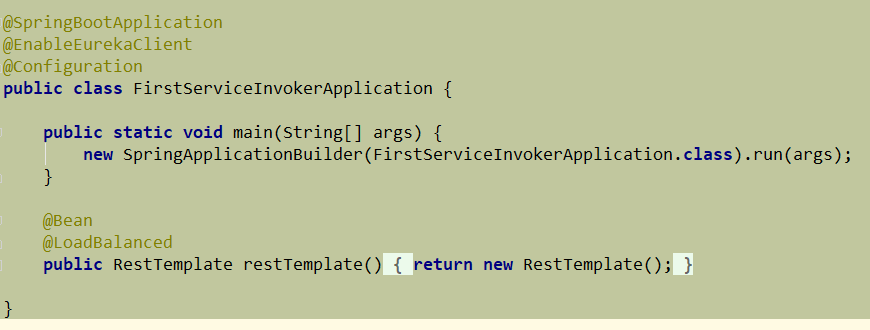
application.properties也大致相似，配置如下 （端口，APP name不一样）：

**server.port**=**9000  
spring.application.name**=**first-service-invoker  
eureka.instance.hostname**=**localhost  
eureka.client.service-url.default-zone**=**http://localhost:8761/eureka/  
spring.zipkin.base-url**=**http://localhost:9411  
spring.sleuth.sampler.percentage**=**1.0**

加一个测试接口：/test 待会要在浏览器调用test接口，然后test接口会去请求first-service-provider的message接口和second-service-provider的message2接口。然后把两个接口的数据返回给用户（浏览器）



这是RestTemplate的配置（注意别漏掉注解哦 否则会有意想不到的惊喜）



**测试程序：**

好啦，所有配置就绪，依次按顺序启动

eureka-server,

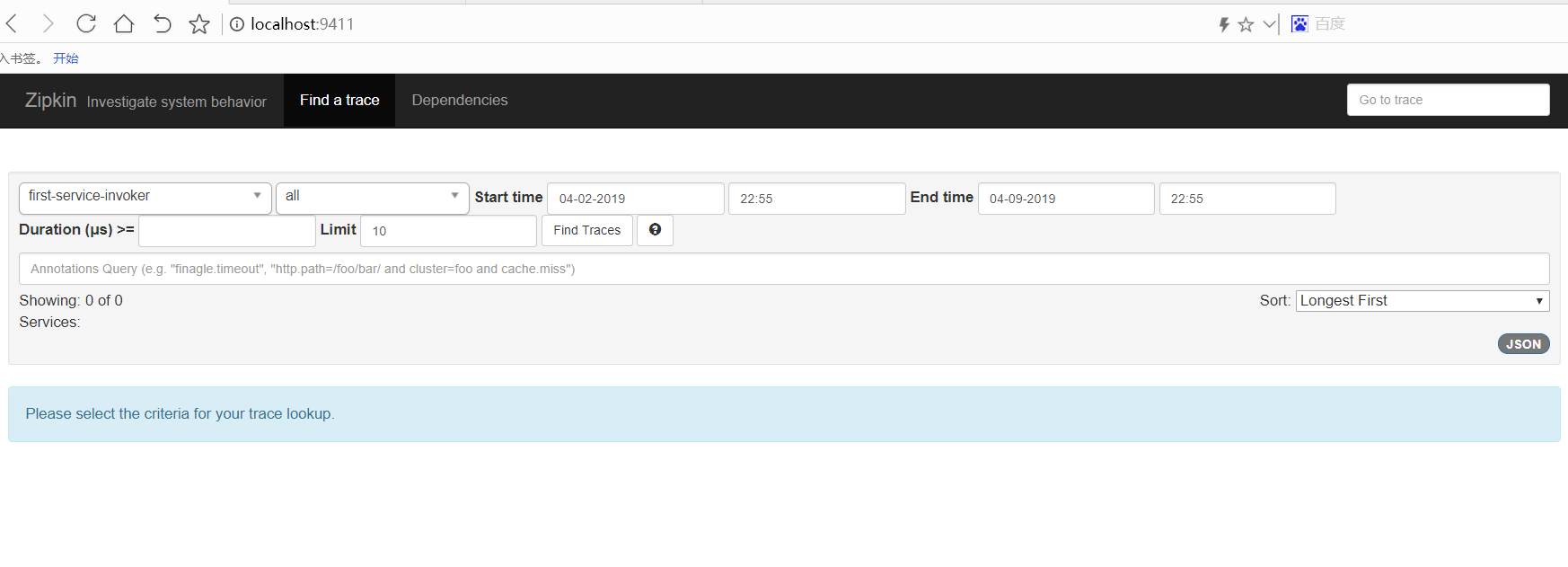
zipkin-server,

first-service-provider,

second-service-provider,

first-service-invoker

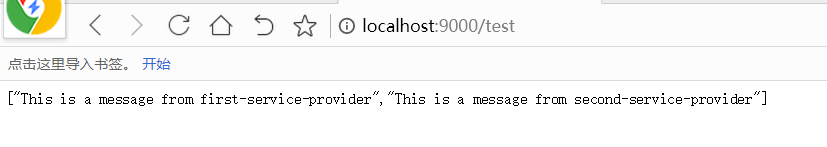
启动成功之后，打开浏览器，访问：localhost:9411 即可看到zipkin的主界面了：



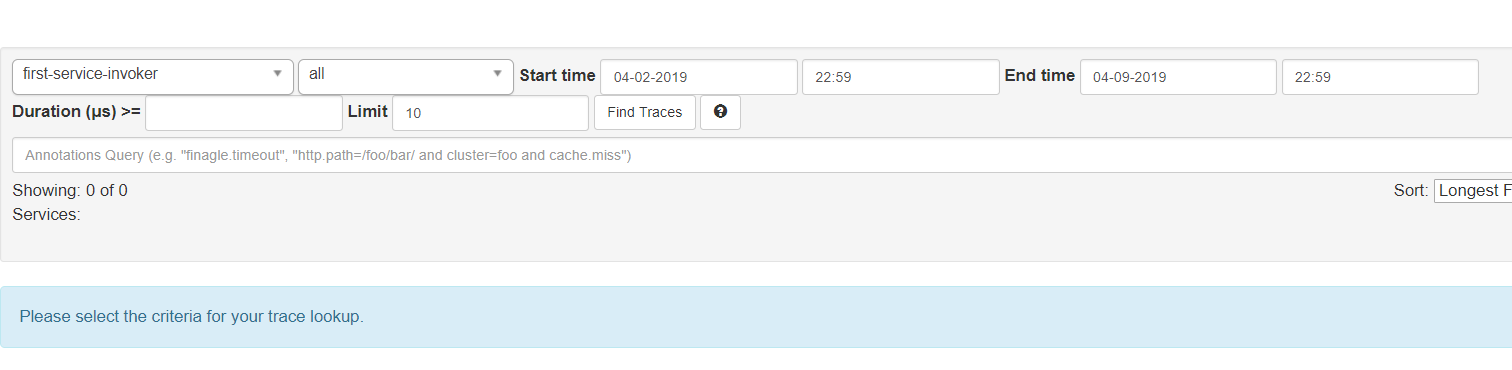
我们再来访问first-service-invoker的test接口，看看sleuth+zipkin如何进行跟踪和跨度。

<http://localhost:9000/test>

可以看到 下面两条消息分别是两个service provider返回过来的，说明调用成功了。

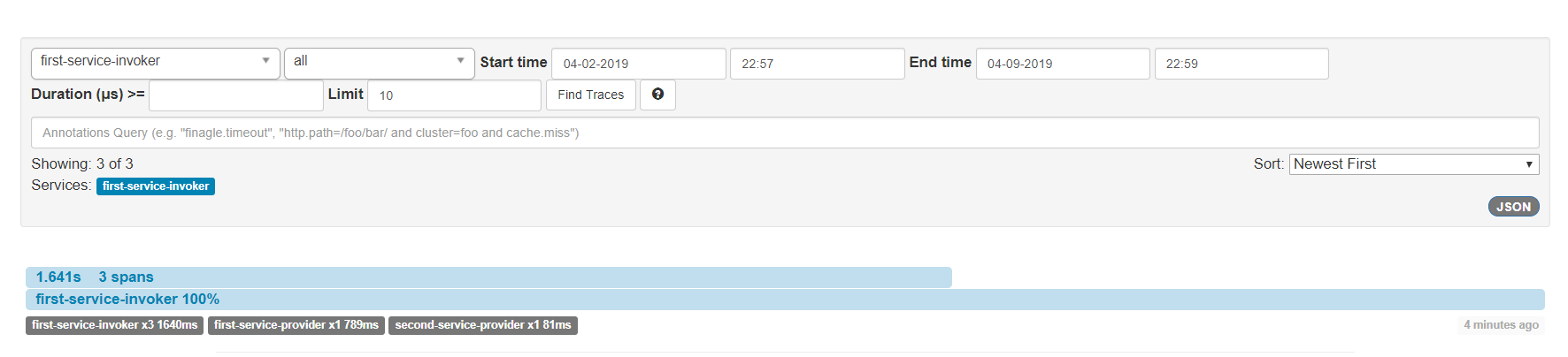


我们再去刷新一下localhost:9411 (zipkin server)



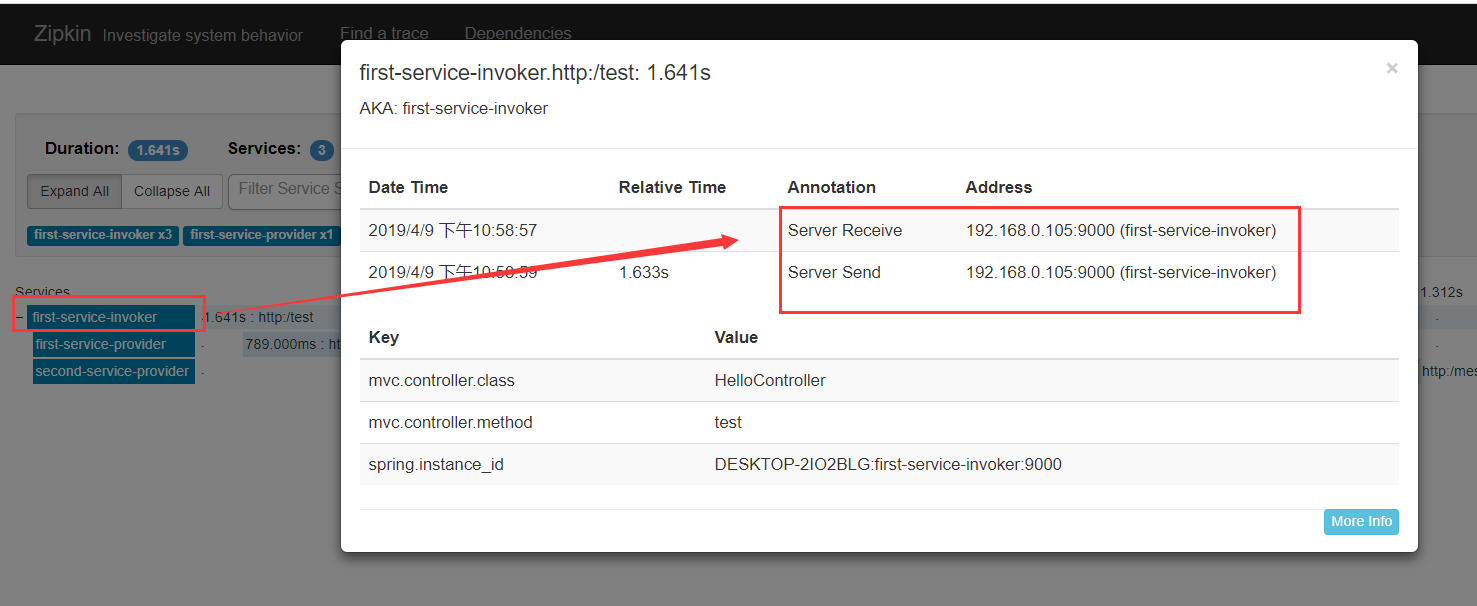
咦，啥都诶有？？别紧张，把时间区间调整一下，默认的区间太短 可能看不到。我们把时间区间调大一点，间隔一个小时或几分钟：

这下就有了

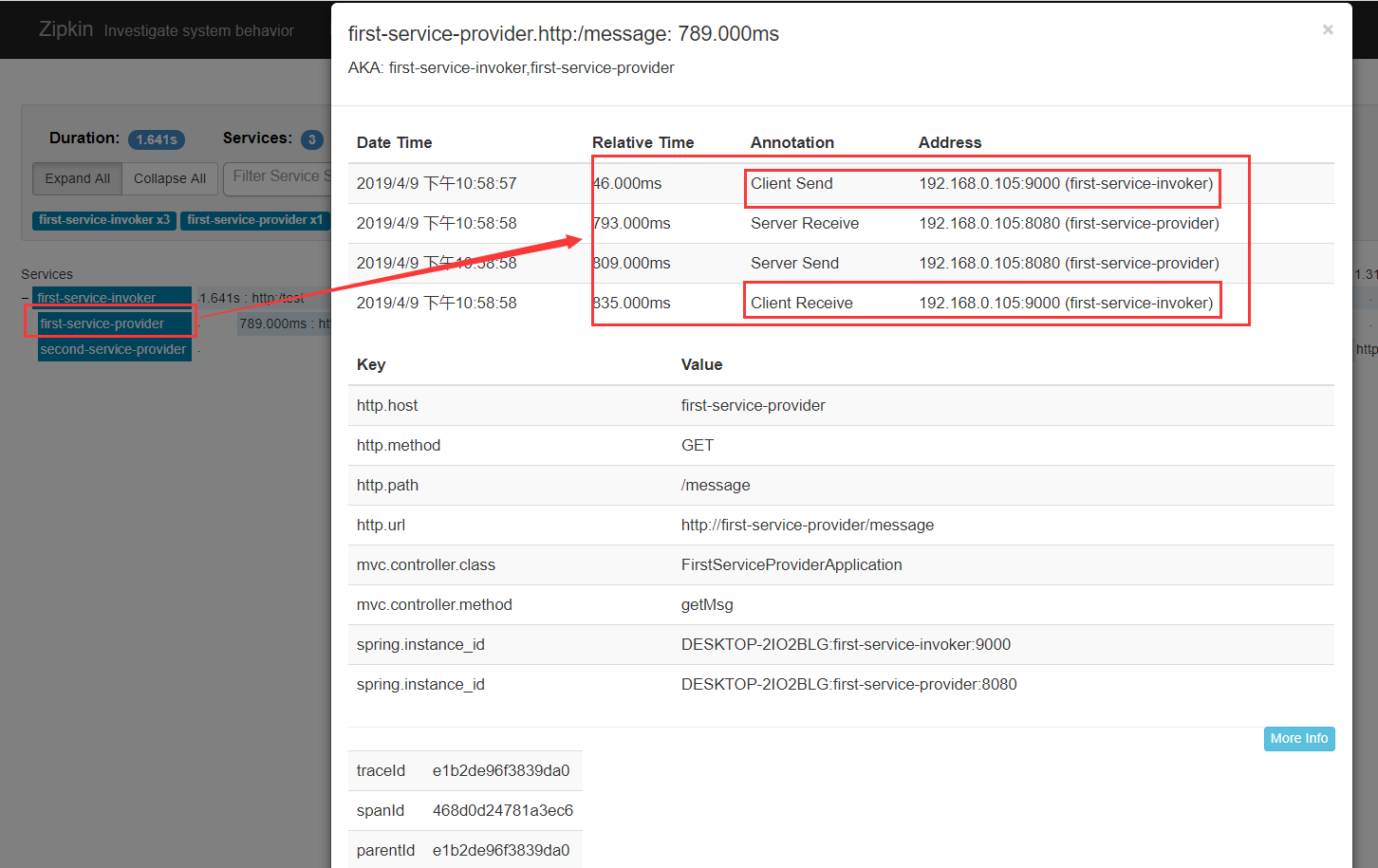


可以发现，有三个跨度，可以点击进去查看详细情况：

点击第一个，可以看到Invoker接收了浏览器的请求，并且响应给浏览器的详细情况和时间。



点击第二个（first-service-provider）可以看到他上面的请求详情，有四个动作，分别是Invoker发送请求，他接收请求，他发送请求，Invoker接收请求：



点击第三个，和第二个类似，也是四个动作，就不贴图了。

主界面上面还有一些过滤条件，可以帮助我们筛选有用的跟踪。右上角有个json图标，点击可以看到json格式的跟踪详情：

[

    {

        "traceId":"e1b2de96f3839da0",

        "id":"e1b2de96f3839da0",

        "name":"http:/test",

        "timestamp":1554821937637000,

        "duration":1640589,

        "annotations":[

            {

                "timestamp":1554821937637000,

                "value":"sr",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "timestamp":1554821939270000,

                "value":"ss",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            }

        ],

        "binaryAnnotations":[

            {

                "key":"mvc.controller.class",

                "value":"HelloController",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"mvc.controller.method",

                "value":"test",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"spring.instance\_id",

                "value":"DESKTOP-2IO2BLG:first-service-invoker:9000",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            }

        ]

    },

    {

        "traceId":"e1b2de96f3839da0",

        "id":"468d0d24781a3ec6",

        "name":"http:/message",

        "parentId":"e1b2de96f3839da0",

        "timestamp":1554821937682000,

        "duration":789000,

        "annotations":[

            {

                "timestamp":1554821937683000,

                "value":"cs",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "timestamp":1554821938430000,

                "value":"sr",

                "endpoint":{

                    "serviceName":"first-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8080

                }

            },

            {

                "timestamp":1554821938446000,

                "value":"ss",

                "endpoint":{

                    "serviceName":"first-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8080

                }

            },

            {

                "timestamp":1554821938472000,

                "value":"cr",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            }

        ],

        "binaryAnnotations":[

            {

                "key":"http.host",

                "value":"first-service-provider",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"http.method",

                "value":"GET",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"http.path",

                "value":"/message",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"http.url",

                "value":"http://first-service-provider/message",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"mvc.controller.class",

                "value":"FirstServiceProviderApplication",

                "endpoint":{

                    "serviceName":"first-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8080

                }

            },

            {

                "key":"mvc.controller.method",

                "value":"getMsg",

                "endpoint":{

                    "serviceName":"first-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8080

                }

            },

            {

                "key":"spring.instance\_id",

                "value":"DESKTOP-2IO2BLG:first-service-invoker:9000",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"spring.instance\_id",

                "value":"DESKTOP-2IO2BLG:first-service-provider:8080",

                "endpoint":{

                    "serviceName":"first-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8080

                }

            }

        ]

    },

    {

        "traceId":"e1b2de96f3839da0",

        "id":"84259e78200f60ea",

        "name":"http:/message2",

        "parentId":"e1b2de96f3839da0",

        "timestamp":1554821938853000,

        "duration":81000,

        "annotations":[

            {

                "timestamp":1554821938853000,

                "value":"cs",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "timestamp":1554821938930000,

                "value":"sr",

                "endpoint":{

                    "serviceName":"second-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8081

                }

            },

            {

                "timestamp":1554821938932000,

                "value":"ss",

                "endpoint":{

                    "serviceName":"second-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8081

                }

            },

            {

                "timestamp":1554821938934000,

                "value":"cr",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            }

        ],

        "binaryAnnotations":[

            {

                "key":"http.host",

                "value":"second-service-provider",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"http.method",

                "value":"GET",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"http.path",

                "value":"/message2",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"http.url",

                "value":"http://second-service-provider/message2",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"mvc.controller.class",

                "value":"SecondServiceProviderApplication",

                "endpoint":{

                    "serviceName":"second-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8081

                }

            },

            {

                "key":"mvc.controller.method",

                "value":"getMsg",

                "endpoint":{

                    "serviceName":"second-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8081

                }

            },

            {

                "key":"spring.instance\_id",

                "value":"DESKTOP-2IO2BLG:first-service-invoker:9000",

                "endpoint":{

                    "serviceName":"first-service-invoker",

                    "ipv4":"192.168.0.105",

                    "port":9000

                }

            },

            {

                "key":"spring.instance\_id",

                "value":"DESKTOP-2IO2BLG:second-service-provider:8081",

                "endpoint":{

                    "serviceName":"second-service-provider",

                    "ipv4":"192.168.0.105",

                    "port":8081

                }

            }

        ]

    }

]