# 目录

# 一 跪求服务启动问题

启动微服务出现的一些警告：

```
WARN [xzl-beg,,,] 16088 --- [ main] c.c.c.ConfigServicePropertySourceLocator : Fetching config from server at : http://localhost:8888
```

解决方案与思路：

> https://blog.csdn.net/Deemo__/article/details/81912316

```
WARN [xzl-beg,,,] 16088 --- [ main] c.c.c.ConfigServicePropertySourceLocator : Could not locate PropertySource: I/O error on GET request for "http://localhost:8888/xzl-beg/default": Connection refused: connect; nested exception is java.net.ConnectException: Connection refused: connect
```

解决方案与思路：

> https://blog.csdn.net/fenglailea/article/details/82783958

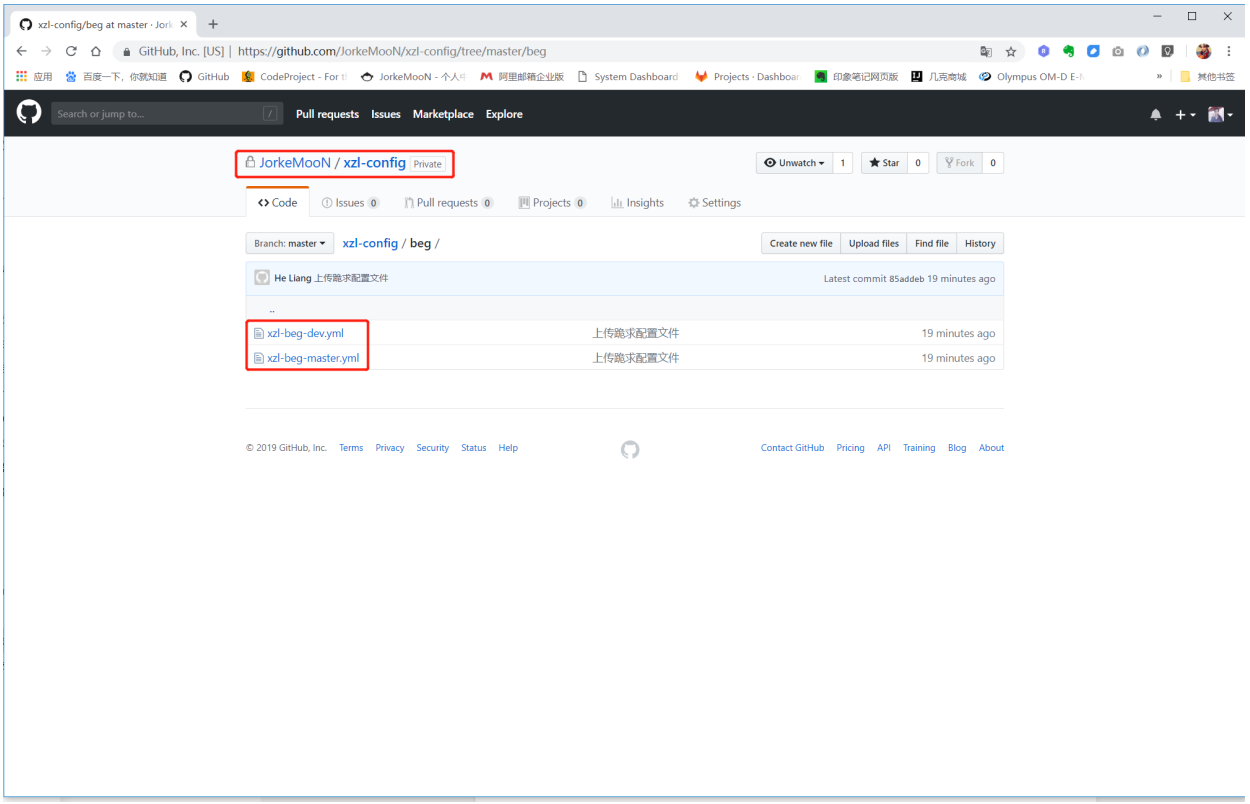
```
WARN [xzl-beg,,,] 16088 --- [ost-startStop-1] c.n.c.sources.URLConfigurationSource : No URLs will be polled as dynamic configuration sources.
```

解决方案与思路：

> https://blog.csdn.net/simpledate/article/details/82777632

# 二 整合远程配置服务

作为配置服务客户端，取github上面的配置文件



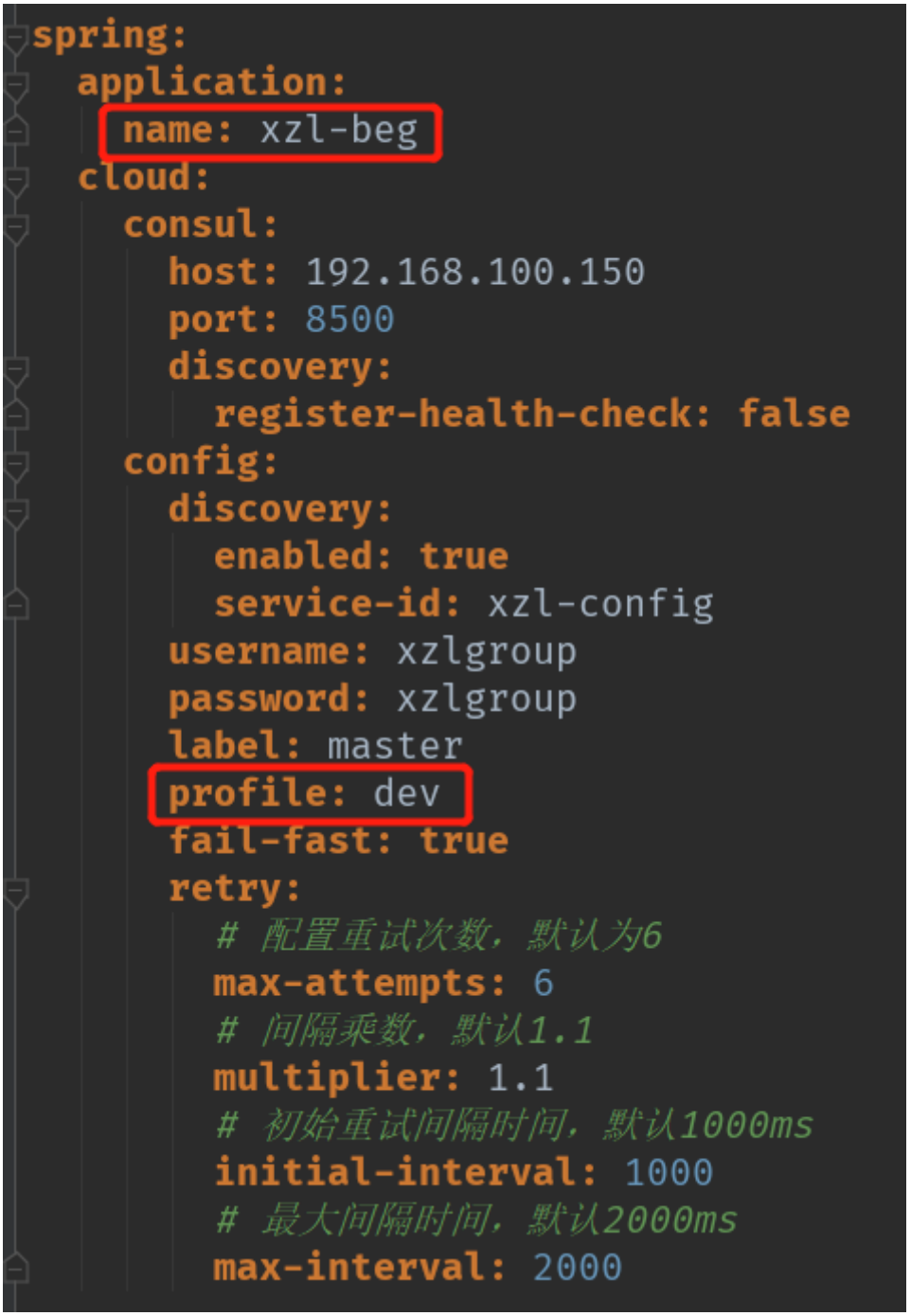

github上面配置文件的路径为

```
https://github.com/JorkeMooN/xzl-config/beg
```

文件夹下面有xzl-beg-dev.yml和xzl-beg-master.yml两个配置文件

命名规则为{spring.application.name}-{spring.cloud.config.profile}.yml

要对应到代码中的配置文件




通过这种方式定位到配置文件后

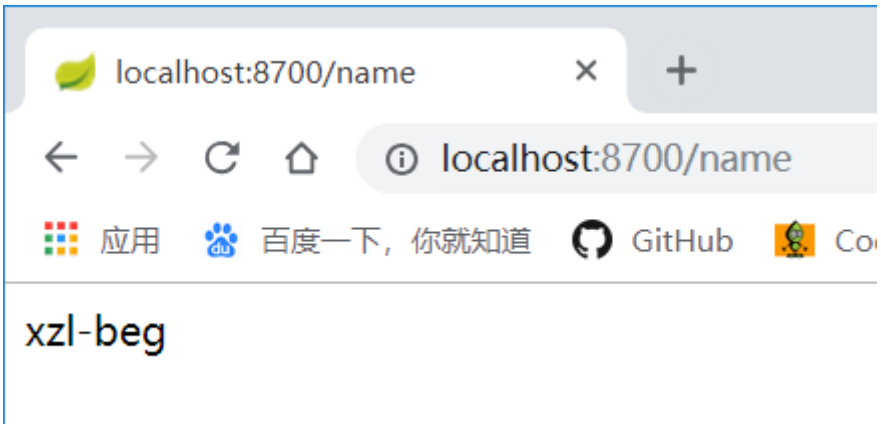
可以使用@Value注解来获取配置的值

例如：

```
@RefreshScope
@RestController
public class ConfigController {
 @Value("${spring.application.name}")
 private String name;

 @GetMapping(value = "/name")
```

```
8   public Object name() {
9   return this.name;
10   }
11  }
```

访问http://localhost:8700/name得到如下




其中@RefreshScope是可以实时更新配置文件信息

操作步骤：

1.通过github修改仓库中的配置文件

2.使用POST方式请求http://localhost:8700/actuator/refresh，手动刷新

刷新返回值：

```
1  [
2   "config.client.version",
3   "spring.application.name"
4  ]
```

通过上述步骤即可实时刷新配置文件

# 三 整合zipkin+sleuth链路跟踪

## 1.引入依赖

```
1  <dependency>
2   <groupId>org.springframework.cloud</groupId>
3   <artifactId>spring-cloud-starter-zipkin</artifactId>
4  </dependency>
```

该引用包含如下两个引用

```
1  <dependencies>
2   <dependency>
```

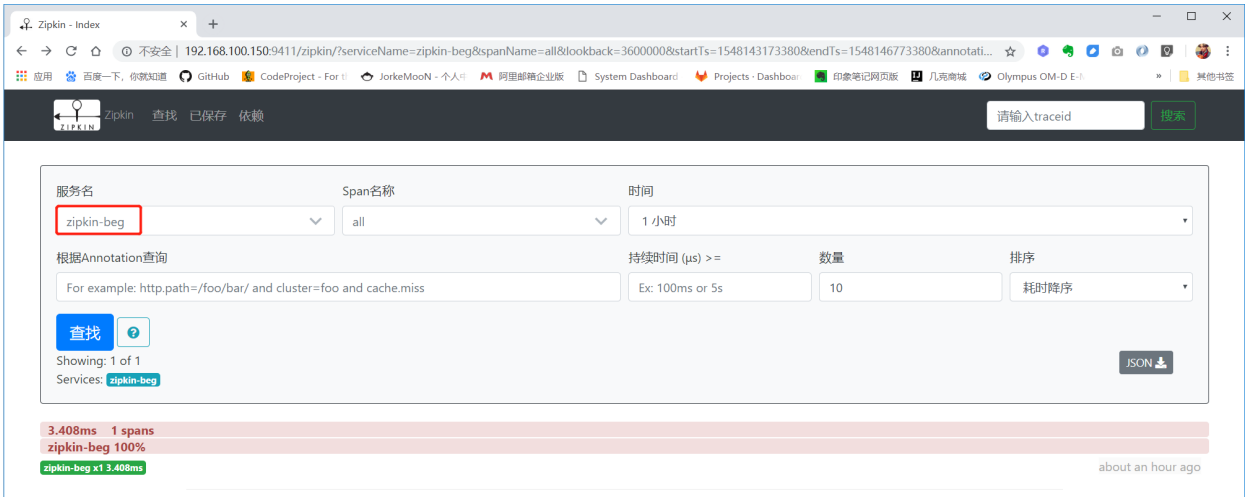
```
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-sleuth</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-sleuth-zipkin</artifactId>
</dependency>
</dependencies>
```

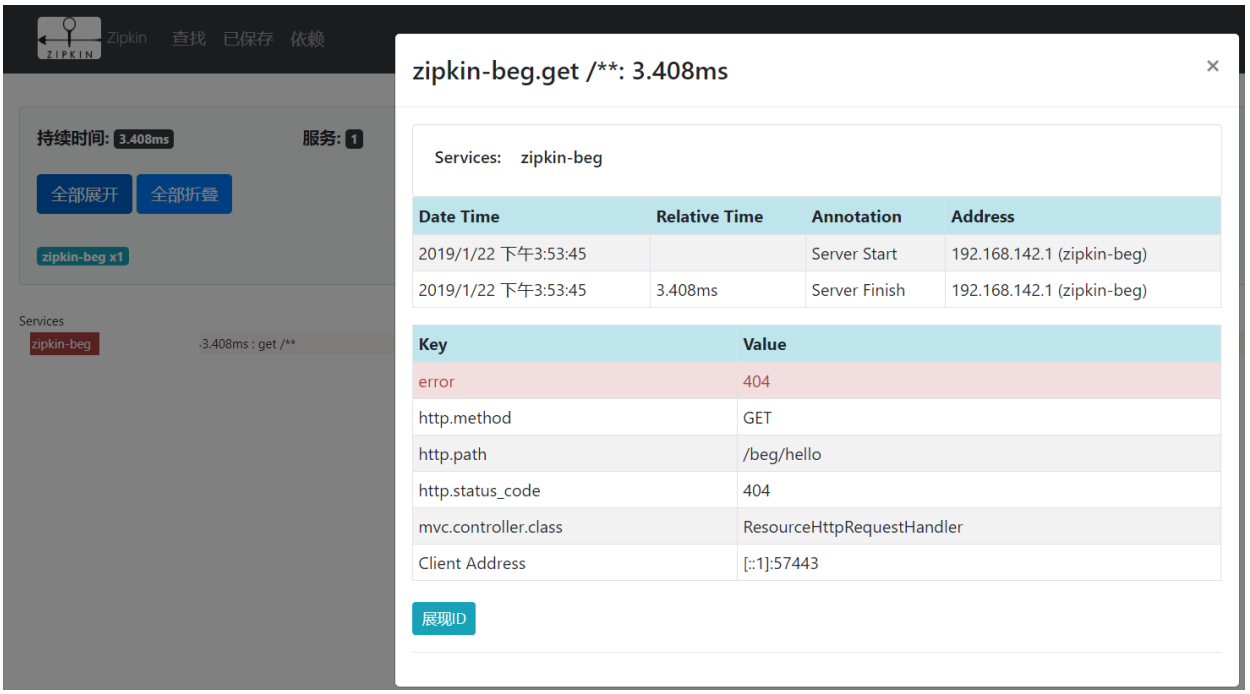
## 2.添加配置

```
spring:
 zipkin:
 service:
 name: zipkin-beg
 base-url: http://192.168.100.150:9411
 compression:
 enabled: true
 sleuth:
 sampler:
  # 采样百分比，默认为0.1，这里配置1，是记录全部的sleuth信息，是为了收集到更多的数据（仅供测试用）。在生产环境需要设置合适的值。
  probability: 1.0
```

> 参考：
> https://www.cnblogs.com/shunyang/p/7011283.html

跟踪到的效果图如下：






# 四 整合zuul网关路由

## 1.zuul项目maven依赖

```
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-consul-discovery</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-zuul</artifactId>
</dependency>
```

## 2.配置文件

```
server:
 port: 8610
 # 网关的session名字，建议每个微服务都单独命名
 servlet:
 session:
 cookie:
```

```yaml
  name: ZUUL_SESSION

spring:
  application:
  name: xzl-zuul
  cloud:
  consul:
  host: 192.168.100.150
  port: 8500
  discovery:
  register-health-check: false

zuul:
  # 忽略框架默认的服务映射路径
  ignored-services: '*'
  # 不忽略框架与权限相关的头信息
  ignore-security-headers: false
  # 不忽略任何头部信息，所有header都转发到下游的资源服务器
  # 所以这里对制定的路由开启自定义敏感头。除了设置为true，也可以设置为空。zuul.sensitiveHeaders= 只是全局设置的做法。不推荐！破坏了默认设置的用意
 # sensitive-headers:
 routes:
 # 所有以/beg开头的请求都转发到xzl-beg应用中
  xzl-beg:
  path: /beg/**
  serviceId: xzl-beg
  # springcloud项目中经过网关zuul转发请求后发生session失效问题，这是由于zuul默认会丢弃原来的session并生成新的session
  sensitiveHeaders: true
```

## 3.配置完成后启动项目

zuul网关需要在 要代理的服务之后启动，放最后启动也没问题

**请求规则：**

| 直接通过xzl-beg服务请求 | 通过zuul网关请求 |
| --- | --- |
| http://localhost:8700/hello | http://localhost:8610/beg/hello |
| http://{原服务IP}:{原服务端口}/{请求地址} | http://{网关IP}:{网关端口}/{配置的前缀}/{请求地址} |

# 五 整合hystrix熔断器，以及dashboard显示

## 1.添加后启动被监控的项目（xzl-beg）

报错如下：

```
java.lang.IllegalStateException: Failed to introspect Class [org.springframework.cloud.netflix.hystrix.HystrixCircuitBreakerConfiguration] from ClassLoader [sun.misc.Launcher$AppClassLoader@18b4aac2]
  at org.springframework.util.ReflectionUtils.getDeclaredMethods(ReflectionUtils.java:659) ~[spring-core-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.util.ReflectionUtils.doWithMethods(ReflectionUtils.java:556) ~[spring-core-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.util.ReflectionUtils.doWithMethods(ReflectionUtils.java:541) ~[spring-core-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.util.ReflectionUtils.getUniqueDeclaredMethods(ReflectionUtils.java:599) ~[spring-core-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.getTypeForFactoryMethod(AbstractAutowireCapableBeanFactory.java:726) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.determineTargetType(AbstractAutowireCapableBeanFactory.java:667) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.predictBeanType(AbstractAutowireCapableBeanFactory.java:635) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.AbstractBeanFactory.isFactoryBean(AbstractBeanFactory.java:1489) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.DefaultListableBeanFactory.doGetBeanNamesForType(DefaultListableBeanFactory.java:420) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.DefaultListableBeanFactory.getBeanNamesForType(DefaultListableBeanFactory.java:390) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.DefaultListableBeanFactory.getBeansOfType(DefaultListableBeanFactory.java:511) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.beans.factory.support.DefaultListableBeanFactory.getBeansOfType(DefaultListableBeanFactory.java:503) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
  at org.springframework.context.support.AbstractApplicationContext.getBeansOfType(AbstractApplicationContext.java:1198) ~[spring-context-5.0.7.RELE
```

```
ASE.jar:5.0.7.RELEASE]
    at org.springframework.boot.SpringApplication.getExitCodeFromMappedException(SpringApplication.java:889) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
    at org.springframework.boot.SpringApplication.getExitCodeFromException(SpringApplication.java:875) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
    at org.springframework.boot.SpringApplication.handleExitCode(SpringApplication.java:861) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
    at org.springframework.boot.SpringApplication.handleRunFailure(SpringApplication.java:810) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
    at org.springframework.boot.SpringApplication.run(SpringApplication.java:338) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
    at org.springframework.boot.SpringApplication.run(SpringApplication.java:1255) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
    at org.springframework.boot.SpringApplication.run(SpringApplication.java:1243) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
    at com.xzlgroup.beg.BegApplication.main(BegApplication.java:15) [classes/:na]
Caused by: java.lang.NoClassDefFoundError: com/netflix/hystrix/contrib/javanica/aop/aspectj/HystrixCommandAspect
    at java.lang.Class.getDeclaredMethods0(Native Method) ~[na:1.8.0_162]
    at java.lang.Class.privateGetDeclaredMethods(Class.java:2701) ~[na:1.8.0_162]
    at java.lang.Class.getDeclaredMethods(Class.java:1975) ~[na:1.8.0_162]
    at org.springframework.util.ReflectionUtils.getDeclaredMethods(ReflectionUtils.java:641) ~[spring-core-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    ... 20 common frames omitted
Caused by: java.lang.ClassNotFoundException: com.netflix.hystrix.contrib.javanica.aop.aspectj.HystrixCommandAspect
    at java.net.URLClassLoader.findClass(URLClassLoader.java:381) ~[na:1.8.0_162]
    at java.lang.ClassLoader.loadClass(ClassLoader.java:424) ~[na:1.8.0_162]
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:338) ~[na:1.8.0_162]
    at java.lang.ClassLoader.loadClass(ClassLoader.java:357) ~[na:1.8.0_162]
    ... 24 common frames omitted
```

解决方案：

明确hystrix的版本号，因为spring cloud没有包含hystrix，所以需要自行制定版本

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

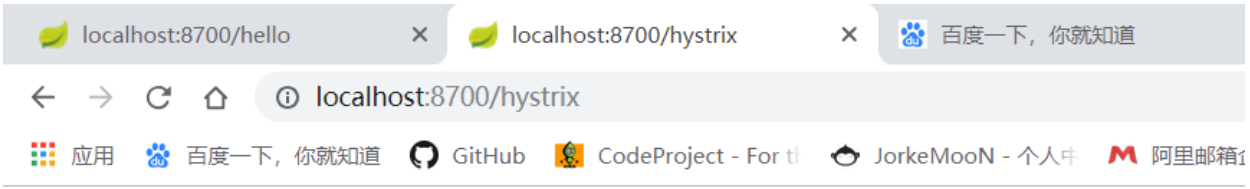
```
  <artifactId>spring-cloud-starter-hystrix</artifactId>
  <version>1.4.4.RELEASE</version>
</dependency>
```

这样即解决了启动报错的问题

## 2.启动成功

但是，hystrix监控还是无法监控到beg服务

请求 http://localhost:8700/hystrix.stream 链接会报404错误




# Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Wed Jan 23 10:19:29 GMT+08:00 2019
There was an unexpected error (type=Not Found, status=404).
No message available

监控后台报错信息：

```
2019-01-23 10:07:15.974 INFO 17000 --- [nio-8620-exec-2] ashboardConfigur
ation$ProxyStreamServlet :

Proxy opening connection to: http://localhost:8700/hystrix.stream


2019-01-23 10:07:15.992 WARN 17000 --- [nio-8620-exec-2] ashboardConfigur
ation$ProxyStreamServlet : Failed opening connection to http://localhost:87
00/hystrix.stream : 404 : HTTP/1.1 404
2019-01-23 10:07:15.992 WARN 17000 --- [nio-8620-exec-4] ashboardConfigur
ation$ProxyStreamServlet : Failed opening connection to http://localhost:87
00/hystrix.stream : 404 : HTTP/1.1 404
```

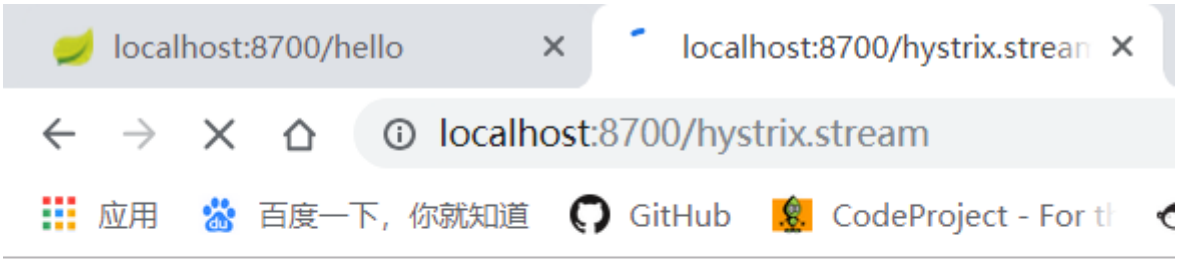
解决方案：

> 参考：
> https://www.cnblogs.com/wangdaijun/p/8891220.html
> https://ask.csdn.net/questions/683294

根据参考文档添加一个bean后，成功

```java
@Configuration
public class HystrixConfig {

 @Bean
 @SuppressWarnings("unchecked")
 public ServletRegistrationBean hystrixMetricsStreamServlet(){
 ServletRegistrationBean registrationBean = new ServletRegistrationBean(new HystrixMetricsStreamServlet());
 registrationBean.addUrlMappings("/hystrix.stream");
 registrationBean.setLoadOnStartup(1);
  registrationBean.setName("HystrixMetricsStreamServlet");
  return registrationBean;
  }

}
```

访问http://localhost:8700/hystrix.stream，效果如下




ping:

ping:

ping:

ping:

ping:

ping:

ping:

ping:

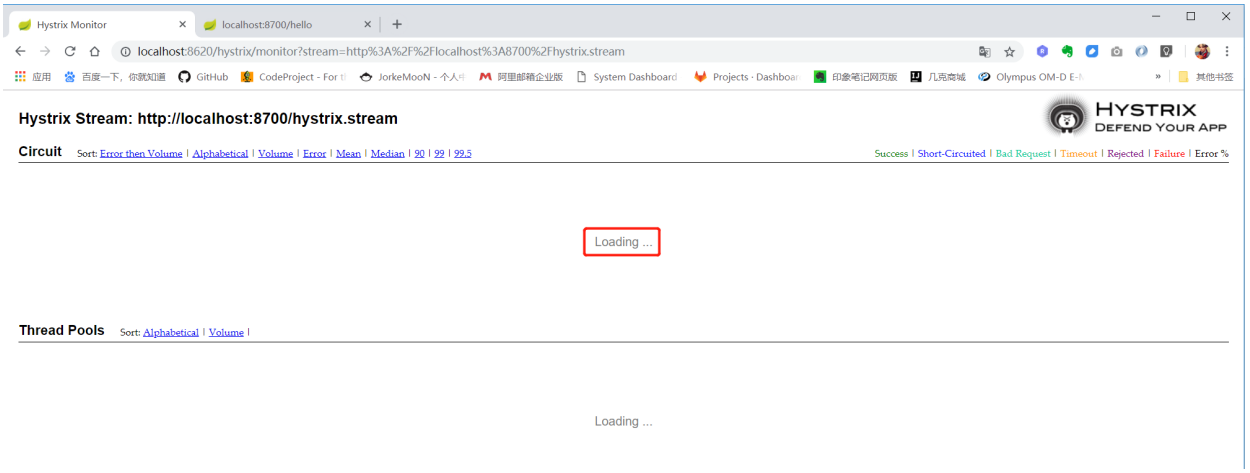
ping:

ping:

ping:

## 3.进入监控页面又有个问题

一直显示 Loading.....



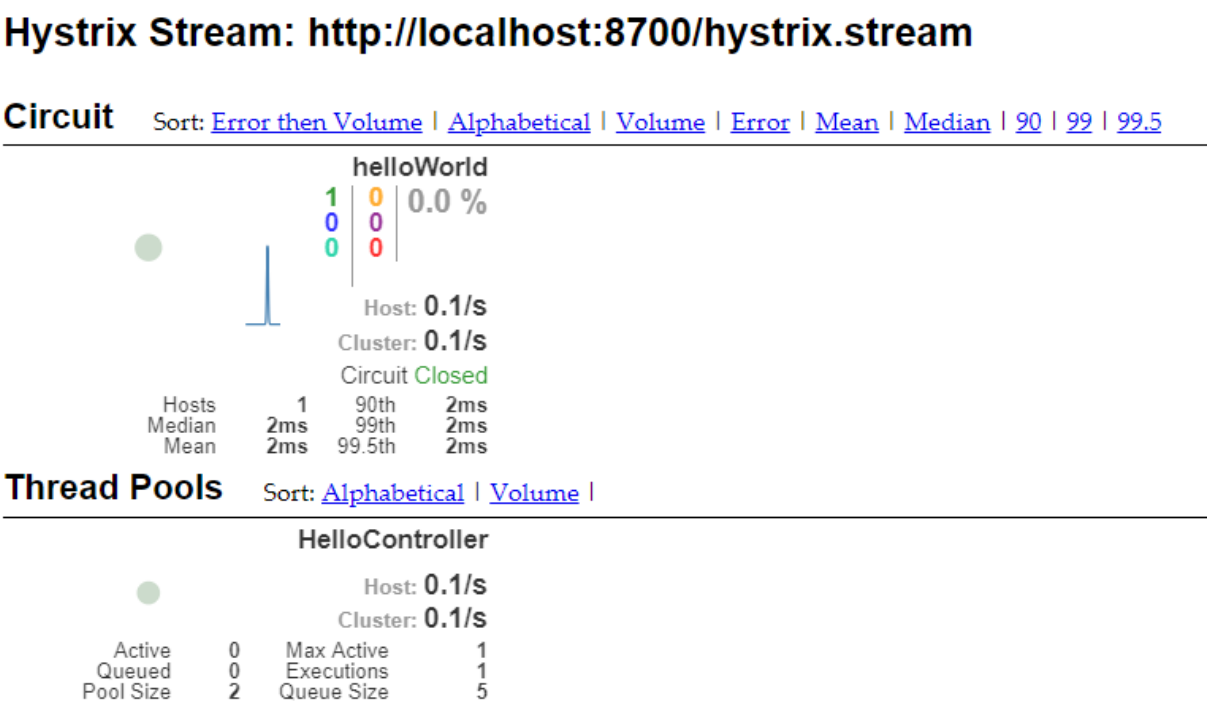

解决方案：

只有开始请求被监控的服务，才能出现监控图

同时，调用的方法需要添加@HystrixCommand

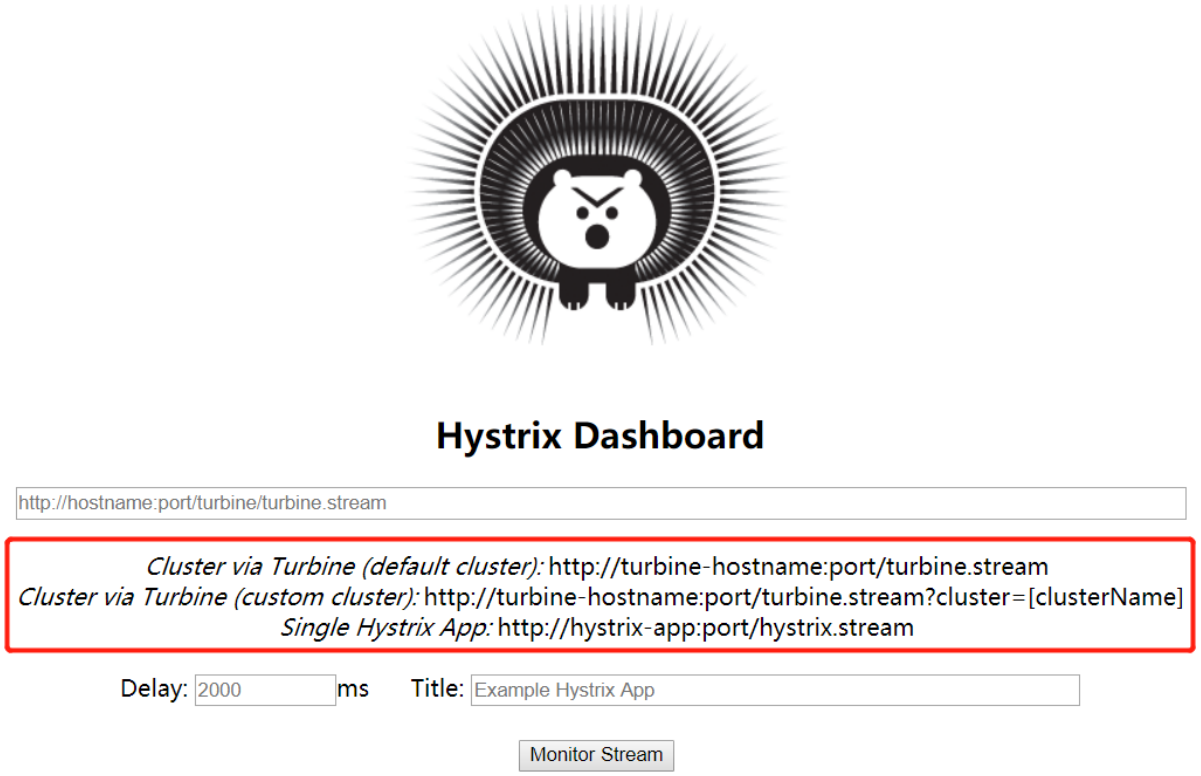
> 参考：
> https://blog.csdn.net/qq_35783540/article/details/81814541

解决完后，监控效果图如下



# 六 整合turbine聚合监控+hystrix dashboard显示

hystrix dashboard首页的提示如下：




```
Cluster via Turbine (default cluster): http://turbine-
  hostname:port/turbine.stream
Cluster via Turbine (custom cluster): http://turbine-
  hostname:port/turbine.stream?cluster=[clusterName]
Single Hystrix App: http://hystrix-app:port/hystrix.stream
```

**这里可以看出**

**1.如果只需要监控单个APP，只使用hystrix dashboard就可以了；**

**2.如果需要监控多个APP，如果使用hystrix dashboard，只能分开监控http://hystrix-app:port/hystrix.stream ，展示效果很差；**

**3.因此，对于监控多个APP，需要添加使用turbine，用来聚合多个APP的监控数据，并且可以自定义监控哪个集群。**

下面开始分开搭建turbine与hystrix dashboard服务

## 1.hystrix dashboard

引入如下依赖：

```
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-consul-discovery</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-hystrix</artifactId>
 </dependency>
 <dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-hystrix-dashboard</artifactId>
 </dependency>
```

配置文件：

```
server:
 port: 8620

spring:
 application:
 name: xzl-hystrix-dashboard
 cloud:
 consul:
 host: 192.168.100.150
  port: 8500
  discovery:
  register-health-check: false
```

## 2.turbine

引入如下依赖：  **(需要排除eureka依赖包)**

```
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

```
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-consul-discovery</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-turbine</artifactId>
  <exclusions>
  <exclusion>
  <groupId>com.netflix.eureka</groupId>
  <artifactId>eureka-core</artifactId>
  </exclusion>
  <exclusion>
  <groupId>com.netflix.eureka</groupId>
  <artifactId>eureka-client</artifactId>
  </exclusion>
  </exclusions>
</dependency>
```

配置文件：

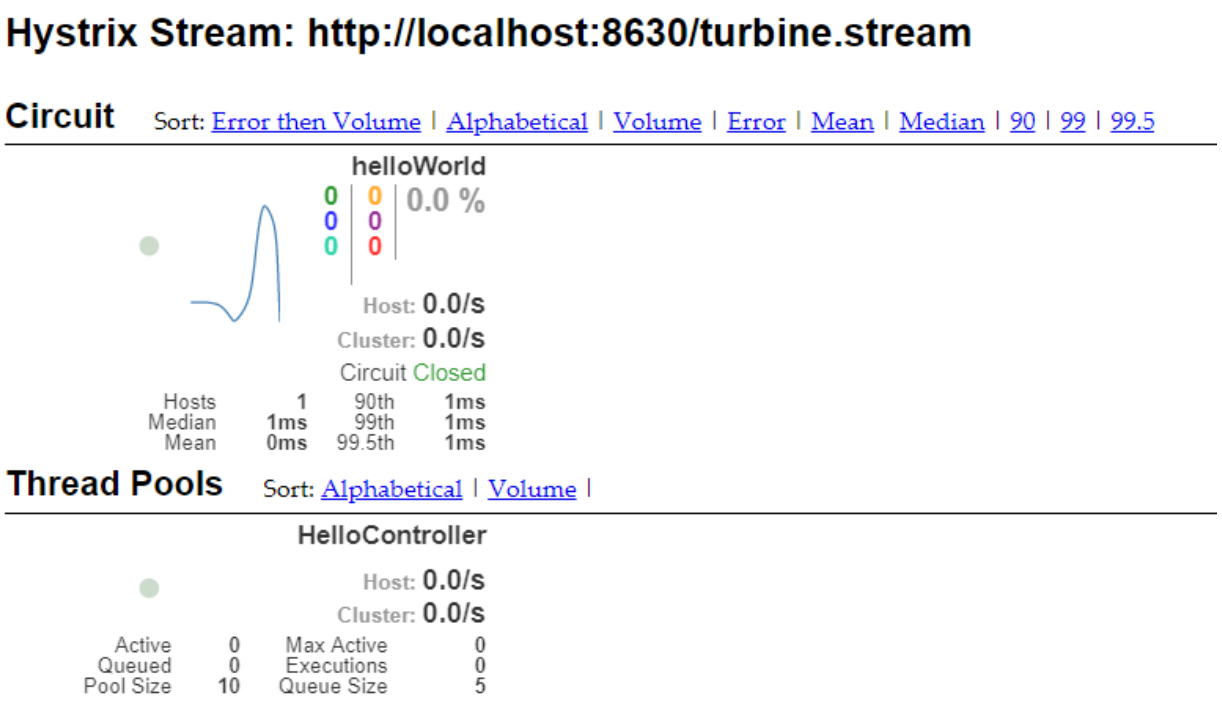
```
server:
 port: 8630

spring:
 application:
 name: xzl-turbine
 cloud:
 consul:
 host: 192.168.100.150
  port: 8500
  discovery:
  register-health-check: false

turbine:
  # 定义所有要监控的微服务
  app-config: xzl-beg
  # 设置监控的表达式，通过此表达式表示要获取监控信息名称
  cluster-name-expression: new String("default")
```

两个服务都搭建成功后启动

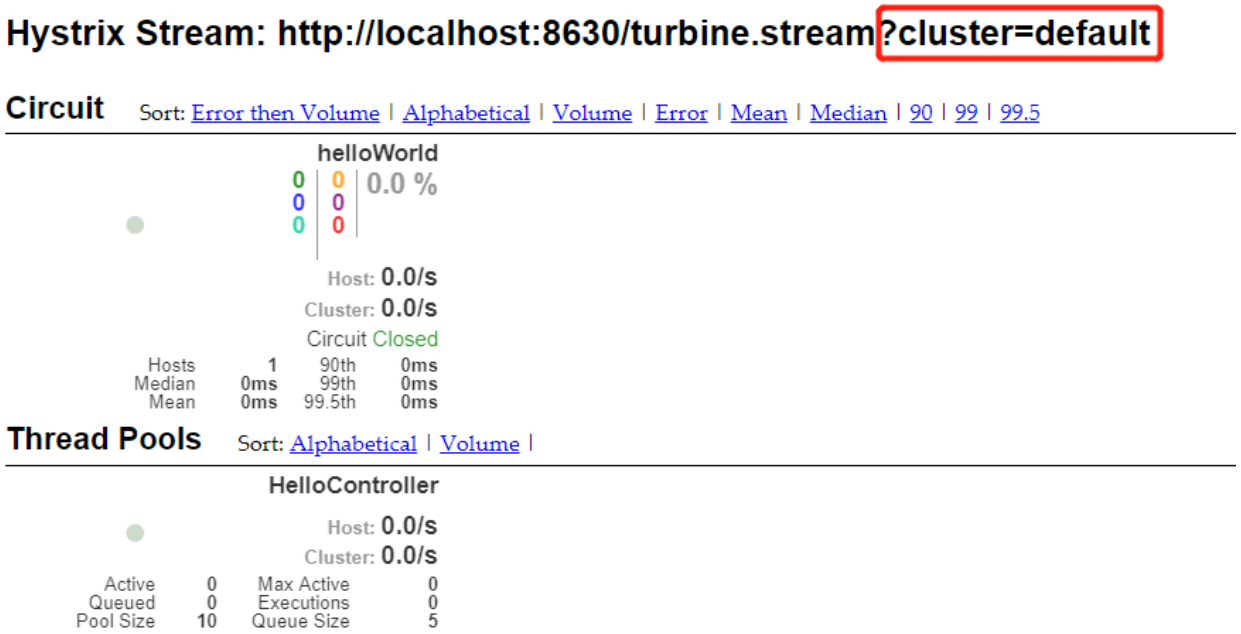
在dashboard中添加监控如下url

```
http://localhost:8630/turbine.stream
```

效果如下：




或者加上**?cluster=default**参数自定义集群名称




turbine聚合后台日志记录：

```
Just added and starting handler tuple: StreamingHandler_909cd779-836d-44e5-a6f9-2c0be86f2c43
```

```
2 2019-01-23 15:09:32.973 INFO 18164 --- [nio-8630-exec-2]
c.n.turbine.data.AggDataFromCluster : Per handler dispacher started for: St
reamingHandler_909cd779-836d-44e5-a6f9-2c0be86f2c43
3 2019-01-23 15:09:32.973 INFO 18164 --- [nio-8630-exec-1] c.n.t.s.TurbineS
treamingConnection : Relevance metrics config: {}
4 2019-01-23 15:09:32.973 INFO 18164 --- [nio-8630-exec-1] c.n.t.monitor.cl
uster.ClusterMonitor : Registering event handler for cluster monitor: Strea
mingHandler_4502734c-3c2e-428e-8d28-e0d0e098e767
5 2019-01-23 15:09:32.974 INFO 18164 --- [nio-8630-exec-1] c.n.t.handler.Tu
rbineDataDispatcher :
6
7 Just added and starting handler tuple: StreamingHandler_4502734c-3c2e-428
e-8d28-e0d0e098e767
8 2019-01-23 15:09:32.974 INFO 18164 --- [nio-8630-exec-2] c.n.t.monitor.cl
uster.ClusterMonitor : All event handlers for cluster monitor: [StaticListe
ner_For_Aggregator, StreamingHandler_909cd779-836d-44e5-a6f9-2c0be86f2c43]
9 2019-01-23 15:09:32.974 INFO 18164 --- [nio-8630-exec-2] c.n.t.monitor.cl
uster.ClusterMonitor : Starting up the cluster monitor for default_agg
10 2019-01-23 15:09:32.974 INFO 18164 --- [nio-8630-exec-1] c.n.turbine.dat
a.AggDataFromCluster : Per handler dispacher started for: StreamingHandler_
4502734c-3c2e-428e-8d28-e0d0e098e767
11 2019-01-23 15:09:32.974 INFO 18164 --- [nio-8630-exec-1] c.n.t.monitor.c
luster.ClusterMonitor : All event handlers for cluster monitor: [StaticList
ener_For_Aggregator, StreamingHandler_4502734c-3c2e-428e-8d28-e0d0e098e767,
StreamingHandler_909cd779-836d-44e5-a6f9-2c0be86f2c43]
12 2019-01-23 15:09:32.974 INFO 18164 --- [nio-8630-exec-1] c.n.t.monitor.c
luster.ClusterMonitor : Starting up the cluster monitor for default_agg
13 2019-01-23 15:10:11.589 INFO 18164 --- [ Timer-0] o.s.c.n.t.CommonsInsta
nceDiscovery : Fetching instance list for apps: [xzl-beg]
14 2019-01-23 15:10:11.589 INFO 18164 --- [ Timer-0] o.s.c.n.t.CommonsInsta
nceDiscovery : Fetching instances for app: xzl-beg
15 2019-01-23 15:10:11.596 INFO 18164 --- [ Timer-0] o.s.c.n.t.CommonsInsta
nceDiscovery : Received instance list for service: xzl-beg, size=1
16 2019-01-23 15:10:11.596 INFO 18164 --- [ Timer-0] c.n.t.discovery.Instan
ceObservable : Retrieved hosts from InstanceDiscovery: 1
17 2019-01-23 15:10:11.597 INFO 18164 --- [ Timer-0] c.n.t.discovery.Instan
ceObservable : Found hosts that have been previously terminated: 0
18 2019-01-23 15:10:11.597 INFO 18164 --- [ Timer-0] c.n.t.discovery.Instan
ceObservable : Hosts up:1, hosts down: 0
19 2019-01-23 15:11:11.589 INFO 18164 --- [ Timer-0] o.s.c.n.t.CommonsInsta
nceDiscovery : Fetching instance list for apps: [xzl-beg]
20 2019-01-23 15:11:11.589 INFO 18164 --- [ Timer-0] o.s.c.n.t.CommonsInsta
nceDiscovery : Fetching instances for app: xzl-beg
21 2019-01-23 15:11:11.594 INFO 18164 --- [ Timer-0] o.s.c.n.t.CommonsInsta
nceDiscovery : Received instance list for service: xzl-beg, size=1
22 2019-01-23 15:11:11.594 INFO 18164 --- [ Timer-0] c.n.t.discovery.Instan
ceObservable : Retrieved hosts from InstanceDiscovery: 1
```

```
2019-01-23 15:11:11.594 INFO 18164 --- [ Timer-0] c.n.t.discovery.InstanceObservable : Found hosts that have been previously terminated: 0
2019-01-23 15:11:11.595 INFO 18164 --- [ Timer-0] c.n.t.discovery.InstanceObservable : Hosts up:1, hosts down: 0
```

# 七 turbine stream+rabbitMQ，解耦合聚合监控数据

## 1.被监控的服务

①添加依赖

被监控的服务需要包含如下依赖：

```
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-hystrix</artifactId>
 <version>1.4.4.RELEASE</version>
</dependency>
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-netflix-hystrix-stream</artifactId>
</dependency>
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-stream-rabbit</artifactId>
</dependency>
```

②添加配置

```
spring:
 rabbitmq:
 host: 192.168.100.150
 port: 5672
 username: geekerx
 password: geekerx
```

重新启动正常。

## 2.turbine服务

①添加依赖

```
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-turbine-stream</artifactId>
 <version>1.4.4.RELEASE</version>
</dependency>
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-stream-rabbit</artifactId>
</dependency>
```

②添加配置

```
spring:
 rabbitmq:
 host: 192.168.100.150
 port: 5672
 username: geekerx
 password: geekerx
```

重新启动报错

日志如下：

```
2019-01-23 15:52:41.490 WARN 7596 --- [ost-startStop-1] o.a.c.loader.WebappClassLoaderBase : The web application [ROOT] appears to have started a thread named [spring.cloud.inetutils] but has failed to stop it. This is very likely to create a memory leak. Stack trace of thread:
 java.net.Inet6AddressImpl.getHostByAddr(Native Method)
 java.net.InetAddress$2.getHostByAddr(InetAddress.java:932)
 java.net.InetAddress.getHostFromNameService(InetAddress.java:617)
 java.net.InetAddress.getHostName(InetAddress.java:559)
 java.net.InetAddress.getHostName(InetAddress.java:531)
 org.springframework.cloud.commons.util.InetUtils$$Lambda$103/1277009227.call(Unknown Source)
 java.util.concurrent.FutureTask.run(FutureTask.java:266)
 java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
 java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
 java.lang.Thread.run(Thread.java:748)
2019-01-23 15:52:41.502 INFO 7596 --- [ main] ConditionEvaluationReportLoggingListener :

Error starting ApplicationContext. To display the conditions report re-run your application with 'debug' enabled.
```

```
2019-01-23 15:52:41.511 ERROR 7596 --- [ main] o.s.boot.SpringApplication : Application run failed

org.springframework.beans.factory.BeanCreationException: Error creating bean with name 'turbineController' defined in org.springframework.cloud.netflix.turbine.stream.TurbineStreamConfiguration: Bean instantiation via factory method failed; nested exception is org.springframework.beans.BeanInstantiationException: Failed to instantiate [org.springframework.cloud.netflix.turbine.stream.TurbineController]: Factory method 'turbineController' threw exception; nested exception is java.lang.BootstrapMethodError: java.lang.NoClassDefFoundError: com/netflix/turbine/aggregator/InstanceKey
    at org.springframework.beans.factory.support.ConstructorResolver.instantiateUsingFactoryMethod(ConstructorResolver.java:590) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.instantiateUsingFactoryMethod(AbstractAutowireCapableBeanFactory.java:1256) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.createBeanInstance(AbstractAutowireCapableBeanFactory.java:1105) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.doCreateBean(AbstractAutowireCapableBeanFactory.java:543) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.createBean(AbstractAutowireCapableBeanFactory.java:503) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.AbstractBeanFactory.lambda$doGetBean$0(AbstractBeanFactory.java:317) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.DefaultSingletonBeanRegistry.getSingleton(DefaultSingletonBeanRegistry.java:222) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.AbstractBeanFactory.doGetBean(AbstractBeanFactory.java:315) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.AbstractBeanFactory.getBean(AbstractBeanFactory.java:199) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.beans.factory.support.DefaultListableBeanFactory.preInstantiateSingletons(DefaultListableBeanFactory.java:760) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.context.support.AbstractApplicationContext.finishBeanFactoryInitialization(AbstractApplicationContext.java:869) ~[spring-context-5.0.7.RELEASE.jar:5.0.7.RELEASE]
    at org.springframework.context.support.AbstractApplicationContext.refresh(AbstractApplicationContext.java:550) ~[spring-context-5.0.7.RELEASE.jar:5.0.7.RELEASE]
```

```
30  at org.springframework.boot.web.servlet.context.ServletWebServerApplica
tionContext.refresh(ServletWebServerApplicationContext.java:140) ~[spring-b
oot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
31  at
org.springframework.boot.SpringApplication.refresh(SpringApplication.java:7
59) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
32  at org.springframework.boot.SpringApplication.refreshContext(SpringAppl
ication.java:395) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
33  at org.springframework.boot.SpringApplication.run(SpringApplication.jav
a:327) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
34  at org.springframework.boot.SpringApplication.run(SpringApplication.jav
a:1255) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
35  at org.springframework.boot.SpringApplication.run(SpringApplication.jav
a:1243) [spring-boot-2.0.3.RELEASE.jar:2.0.3.RELEASE]
36  at
com.xzlgroup.turbine.TurbineApplication.main(TurbineApplication.java:14) [c
lasses/:na]
37 Caused by: org.springframework.beans.BeanInstantiationException: Failed
to instantiate [org.springframework.cloud.netflix.turbine.stream.TurbineCon
troller]: Factory method 'turbineController' threw exception; nested except
ion is java.lang.BootstrapMethodError: java.lang.NoClassDefFoundError:
com/netflix/turbine/aggregator/InstanceKey
38  at org.springframework.beans.factory.support.SimpleInstantiationStrateg
y.instantiate(SimpleInstantiationStrategy.java:185) ~[spring-beans-5.0.7.RE
LEASE.jar:5.0.7.RELEASE]
39  at org.springframework.beans.factory.support.ConstructorResolver.instan
tiateUsingFactoryMethod(ConstructorResolver.java:582) ~[spring-beans-
5.0.7.RELEASE.jar:5.0.7.RELEASE]
40  ... 18 common frames omitted
41 Caused by: java.lang.BootstrapMethodError: java.lang.NoClassDefFoundErro
r: com/netflix/turbine/aggregator/InstanceKey
42  at org.springframework.cloud.netflix.turbine.stream.TurbineController.
<init>(TurbineController.java:44) ~[spring-cloud-netflix-turbine-stream-
2.0.0.RELEASE.jar:2.0.0.RELEASE]
43  at org.springframework.cloud.netflix.turbine.stream.TurbineStreamConfig
uration.turbineController(TurbineStreamConfiguration.java:48) ~[spring-clou
d-netflix-turbine-stream-2.0.0.RELEASE.jar:2.0.0.RELEASE]
44  at org.springframework.cloud.netflix.turbine.stream.TurbineStreamConfig
uration$$EnhancerBySpringCGLIB$$927ee1c6.CGLIB$turbineController$1(<generat
ed>) ~[spring-cloud-netflix-turbine-stream-2.0.0.RELEASE.jar:2.0.0.RELEASE]
45  at org.springframework.cloud.netflix.turbine.stream.TurbineStreamConfig
uration$$EnhancerBySpringCGLIB$$927ee1c6$$FastClassBySpringCGLIB$$f964cfc5.
nvoke(<generated>) ~[spring-cloud-netflix-turbine-stream-
2.0.0.RELEASE.jar:2.0.0.RELEASE]
46  at
org.springframework.cglib.proxy.MethodProxy.invokeSuper(MethodProxy.java:22
8) ~[spring-core-5.0.7.RELEASE.jar:5.0.7.RELEASE]
```

```
47  at org.springframework.context.annotation.ConfigurationClassEnhancer$BeanMethodInterceptor.intercept(ConfigurationClassEnhancer.java:361) ~[spring-context-5.0.7.RELEASE.jar:5.0.7.RELEASE]
48  at org.springframework.cloud.netflix.turbine.stream.TurbineStreamConfiguration$$EnhancerBySpringCGLIB$$927ee1c6.turbineController(<generated>) ~[spring-cloud-netflix-turbine-stream-2.0.0.RELEASE.jar:2.0.0.RELEASE]
49  at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) ~[na:1.8.0_162]
50  at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62) ~[na:1.8.0_162]
51  at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) ~[na:1.8.0_162]
52  at java.lang.reflect.Method.invoke(Method.java:498) ~[na:1.8.0_162]
53  at org.springframework.beans.factory.support.SimpleInstantiationStrategy.instantiate(SimpleInstantiationStrategy.java:154) ~[spring-beans-5.0.7.RELEASE.jar:5.0.7.RELEASE]
54  ... 19 common frames omitted
55 Caused by: java.lang.NoClassDefFoundError: com/netflix/turbine/aggregator/InstanceKey
56  ... 31 common frames omitted
57 Caused by: java.lang.ClassNotFoundException: com.netflix.turbine.aggregator.InstanceKey
58  at java.net.URLClassLoader.findClass(URLClassLoader.java:381) ~[na:1.8.0_162]
59  at java.lang.ClassLoader.loadClass(ClassLoader.java:424) ~[na:1.8.0_162]
60  at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:338) ~[na:1.8.0_162]
61  at java.lang.ClassLoader.loadClass(ClassLoader.java:357) ~[na:1.8.0_162]
62  ... 31 common frames omitted
```

解决方案：

修改pom引用，保证jar包库中，com.netflix.turbine:turbine-core版本为2.0.0-DP.2即可

还会有如下错：

```
1 SLF4J: Class path contains multiple SLF4J bindings.
2 SLF4J: Found binding in [jar:file:/F:/maven-dependencies/ch/qos/logback/logback-classic/1.2.3/logback-classic-1.2.3.jar!/org/slf4j/impl/StaticLoggerBinder.class]
3 SLF4J: Found binding in [jar:file:/F:/maven-dependencies/org/slf4j/slf4j-simple/1.7.25/slf4j-simple-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
```

```
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [ch.qos.logback.classic.util.ContextSelectorStaticBinder]
```

在maven依赖中添加排除即可

最终项目引用如下即可：**（最完整版）**

```
<!-- 保证有turbine.app-config和turbine.cluster-name-expression配置 -->
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-turbine</artifactId>
 <version>1.4.4.RELEASE</version>
 <exclusions>
 <exclusion>
 <groupId>com.netflix.eureka</groupId>
 <artifactId>eureka-core</artifactId>
 </exclusion>
 <exclusion>
 <groupId>com.netflix.eureka</groupId>
 <artifactId>eureka-client</artifactId>
 </exclusion>
 </exclusions>
</dependency>
<!-- Spring Cloud的RabbitMQ的实现，实际上包装了spring-cloud-stater-turbine-stream和spring-cloud-starter-stream-rabbitmq -->
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-turbine-amqp</artifactId>
 <version>1.4.4.RELEASE</version>
 <exclusions>
 <exclusion>
 <groupId>com.netflix.eureka</groupId>
 <artifactId>eureka-core</artifactId>
 </exclusion>
 <exclusion>
 <groupId>com.netflix.eureka</groupId>
 <artifactId>eureka-client</artifactId>
 </exclusion>
 </exclusions>
</dependency>
```

```
<!-- 加入此依赖，避免 Caused by: java.lang.ClassNotFoundException: com.netflix.turbine.aggregator.InstanceKey -->
<dependency>
 <groupId>com.netflix.turbine</groupId>
 <artifactId>turbine-core</artifactId>
 <version>2.0.0-DP.2</version>
 <!-- 避免 SLF4J: Class path contains multiple SLF4J bindings. -->
 <exclusions>
 <exclusion>
 <groupId>org.slf4j</groupId>
 <artifactId>slf4j-simple</artifactId>
 </exclusion>
 </exclusions>
</dependency>
```

再次启动，报错如下：

这次是跟rabbitMQ链接有关

```
2019-01-23 16:10:17.964 WARN 19608 --- [on(8)-127.0.0.1] o.s.b.a.amqp.RabbitHealthIndicator : Rabbit health check failed

org.springframework.amqp.AmqpIOException: java.io.IOException
  at org.springframework.amqp.rabbit.support.RabbitExceptionTranslator.convertRabbitAccessException(RabbitExceptionTranslator.java:71) ~[spring-rabbit-2.0.4.RELEASE.jar:2.0.4.RELEASE]
  at org.springframework.amqp.rabbit.connection.AbstractConnectionFactory.createBareConnection(AbstractConnectionFactory.java:476) ~[spring-rabbit-2.0.4.RELEASE.jar:2.0.4.RELEASE]
  at org.springframework.amqp.rabbit.connection.CachingConnectionFactory.createConnection(CachingConnectionFactory.java:614) ~[spring-rabbit-2.0.4.RELEASE.jar:2.0.4.RELEASE]
  at org.springframework.amqp.rabbit.connection.ConnectionFactoryUtils.createConnection(ConnectionFactoryUtils.java:240) ~[spring-rabbit-2.0.4.RELEASE.jar:2.0.4.RELEASE]
  at org.springframework.amqp.rabbit.core.RabbitTemplate.doExecute(RabbitTemplate.java:1797) ~[spring-rabbit-2.0.4.RELEASE.jar:2.0.4.RELEASE]
  at org.springframework.amqp.rabbit.core.RabbitTemplate.execute(RabbitTemplate.java:1771) ~[spring-rabbit-2.0.4.RELEASE.jar:2.0.4.RELEASE]
  at org.springframework.amqp.rabbit.core.RabbitTemplate.execute(RabbitTemplate.java:1752) ~[spring-rabbit-2.0.4.RELEASE.jar:2.0.4.RELEASE]
  at org.springframework.boot.actuate.amqp.RabbitHealthIndicator.getVersion(RabbitHealthIndicator.java:48) ~[spring-boot-actuator-2.0.3.RELEASE.jar:2.0.3.RELEASE]
```

```
12  at org.springframework.boot.actuate.amqp.RabbitHealthIndicator.doHealth
Check(RabbitHealthIndicator.java:44) ~[spring-boot-actuator-2.0.3.RELEASE.j
ar:2.0.3.RELEASE]
13  at org.springframework.boot.actuate.health.AbstractHealthIndicator.heal
th(AbstractHealthIndicator.java:84) ~[spring-boot-actuator-2.0.3.RELEASE.ja
r:2.0.3.RELEASE]
14  at org.springframework.boot.actuate.health.CompositeHealthIndicator.hea
lth(CompositeHealthIndicator.java:68) [spring-boot-actuator-2.0.3.RELEASE.j
ar:2.0.3.RELEASE]
15  at org.springframework.boot.actuate.health.HealthEndpoint.health(Health
Endpoint.java:47) [spring-boot-actuator-2.0.3.RELEASE.jar:2.0.3.RELEASE]
16  at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) ~
[na:1.8.0_162]
17  at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:6
2) ~[na:1.8.0_162]
18  at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcce
ssorImpl.java:43) ~[na:1.8.0_162]
19  at java.lang.reflect.Method.invoke(Method.java:498) ~[na:1.8.0_162]
20  at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtil
s.java:223) [spring-core-5.0.7.RELEASE.jar:5.0.7.RELEASE]
21  at org.springframework.boot.actuate.endpoint.invoke.reflect.ReflectiveO
perationInvoker.invoke(ReflectiveOperationInvoker.java:76) [spring-boot-act
uator-2.0.3.RELEASE.jar:2.0.3.RELEASE]
22  at org.springframework.boot.actuate.endpoint.annotation.AbstractDiscove
redOperation.invoke(AbstractDiscoveredOperation.java:61) [spring-boot-actua
tor-2.0.3.RELEASE.jar:2.0.3.RELEASE]
23  at org.springframework.boot.actuate.endpoint.jmx.EndpointMBean.invoke(E
ndpointMBean.java:126) [spring-boot-actuator-2.0.3.RELEASE.jar:2.0.3.RELEAS
E]
24  at org.springframework.boot.actuate.endpoint.jmx.EndpointMBean.invoke(E
ndpointMBean.java:99) [spring-boot-actuator-
2.0.3.RELEASE.jar:2.0.3.RELEASE]
25  at com.sun.jmx.interceptor.DefaultMBeanServerInterceptor.invoke(Default
MBeanServerInterceptor.java:819) [na:1.8.0_162]
26  at com.sun.jmx.mbeanserver.JmxMBeanServer.invoke(JmxMBeanServer.java:80
1) [na:1.8.0_162]
27  at javax.management.remote.rmi.RMIConnectionImpl.doOperation(RMIConnect
ionImpl.java:1468) [na:1.8.0_162]
28  at javax.management.remote.rmi.RMIConnectionImpl.access$300(RMIConnecti
onImpl.java:76) [na:1.8.0_162]
29  at javax.management.remote.rmi.RMIConnectionImpl$PrivilegedOperation.ru
n(RMIConnectionImpl.java:1309) [na:1.8.0_162]
30  at
javax.management.remote.rmi.RMIConnectionImpl.doPrivilegedOperation(RMIConn
ectionImpl.java:1401) [na:1.8.0_162]
```

```
31  at javax.management.remote.rmi.RMIConnectionImpl.invoke(RMIConnectionImpl.java:829) [na:1.8.0_162]
32  at sun.reflect.GeneratedMethodAccessor91.invoke(Unknown Source) ~[na:na]
33  at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) ~[na:1.8.0_162]
34  at java.lang.reflect.Method.invoke(Method.java:498) ~[na:1.8.0_162]
35  at sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:361) [na:1.8.0_162]
36  at sun.rmi.transport.Transport$1.run(Transport.java:200) [na:1.8.0_162]
37  at sun.rmi.transport.Transport$1.run(Transport.java:197) [na:1.8.0_162]
38  at java.security.AccessController.doPrivileged(Native Method) [na:1.8.0_162]
39  at sun.rmi.transport.Transport.serviceCall(Transport.java:196) [na:1.8.0_162]
40  at sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:568) [na:1.8.0_162]
41  at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run0(TCPTransport.java:826) [na:1.8.0_162]
42  at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.lambda$run$0(TCPTransport.java:683) [na:1.8.0_162]
43  at java.security.AccessController.doPrivileged(Native Method) [na:1.8.0_162]
44  at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTransport.java:682) [na:1.8.0_162]
45  at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149) ~[na:1.8.0_162]
46  at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624) ~[na:1.8.0_162]
47  at java.lang.Thread.run(Thread.java:748) ~[na:1.8.0_162]
48 Caused by: java.io.IOException: null
49  at com.rabbitmq.client.impl.AMQChannel.wrap(AMQChannel.java:126) ~[amqp-client-5.1.2.jar:5.1.2]
50  at com.rabbitmq.client.impl.AMQChannel.wrap(AMQChannel.java:122) ~[amqp-client-5.1.2.jar:5.1.2]
51  at com.rabbitmq.client.impl.AMQChannel.exnWrappingRpc(AMQChannel.java:144) ~[amqp-client-5.1.2.jar:5.1.2]
52  at com.rabbitmq.client.impl.AMQConnection.start(AMQConnection.java:390) ~[amqp-client-5.1.2.jar:5.1.2]
53  at com.rabbitmq.client.ConnectionFactory.newConnection(ConnectionFactory.java:957) ~[amqp-client-5.1.2.jar:5.1.2]
54  at com.rabbitmq.client.ConnectionFactory.newConnection(ConnectionFactory.java:907) ~[amqp-client-5.1.2.jar:5.1.2]
```

```
55  at com.rabbitmq.client.ConnectionFactory.newConnection(ConnectionFactor
y.java:847) ~[amqp-client-5.1.2.jar:5.1.2]
56  at
org.springframework.amqp.rabbit.connection.AbstractConnectionFactory.create
BareConnection(AbstractConnectionFactory.java:449) ~[spring-rabbit-2.0.4.RE
LEASE.jar:2.0.4.RELEASE]
57  ... 42 common frames omitted
58 Caused by: com.rabbitmq.client.ShutdownSignalException: connection
error; protocol method: #method<connection.close>(reply-code=530, reply-tex
t=NOT_ALLOWED - access to vhost '/' refused for user 'geekerx', class-
id=10, method-id=40)
59  at
com.rabbitmq.utility.ValueOrException.getValue(ValueOrException.java:66) ~
[amqp-client-5.1.2.jar:5.1.2]
60  at com.rabbitmq.utility.BlockingValueOrException.uninterruptibleGetValu
e(BlockingValueOrException.java:36) ~[amqp-client-5.1.2.jar:5.1.2]
61  at
com.rabbitmq.client.impl.AMQChannel$BlockingRpcContinuation.getReply(AMQCha
nnel.java:494) ~[amqp-client-5.1.2.jar:5.1.2]
62  at com.rabbitmq.client.impl.AMQChannel.privateRpc(AMQChannel.java:288)
~[amqp-client-5.1.2.jar:5.1.2]
63  at com.rabbitmq.client.impl.AMQChannel.exnWrappingRpc(AMQChannel.java:1
38) ~[amqp-client-5.1.2.jar:5.1.2]
64  ... 47 common frames omitted
```

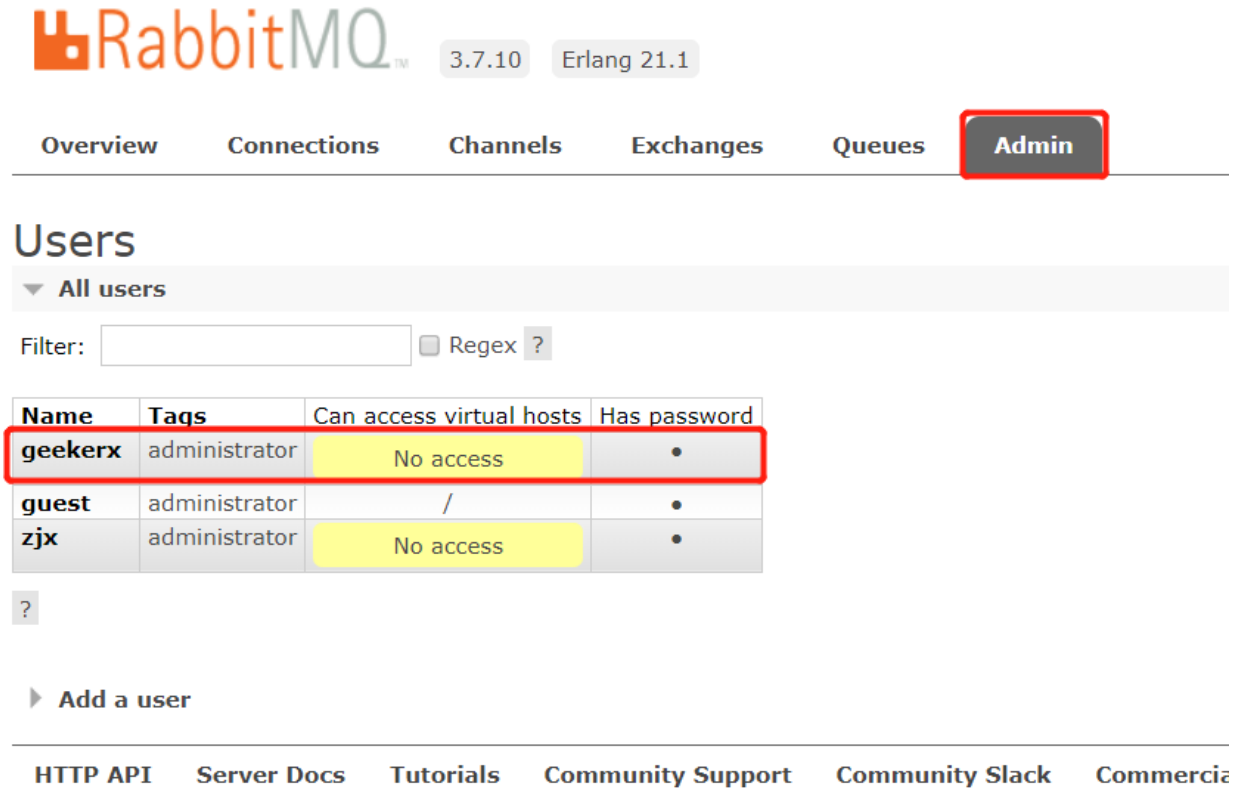
关键内容：

```
1 reply-text=NOT_ALLOWED - access to vhost '/' refused for user 'geekerx'
```

解决方案：

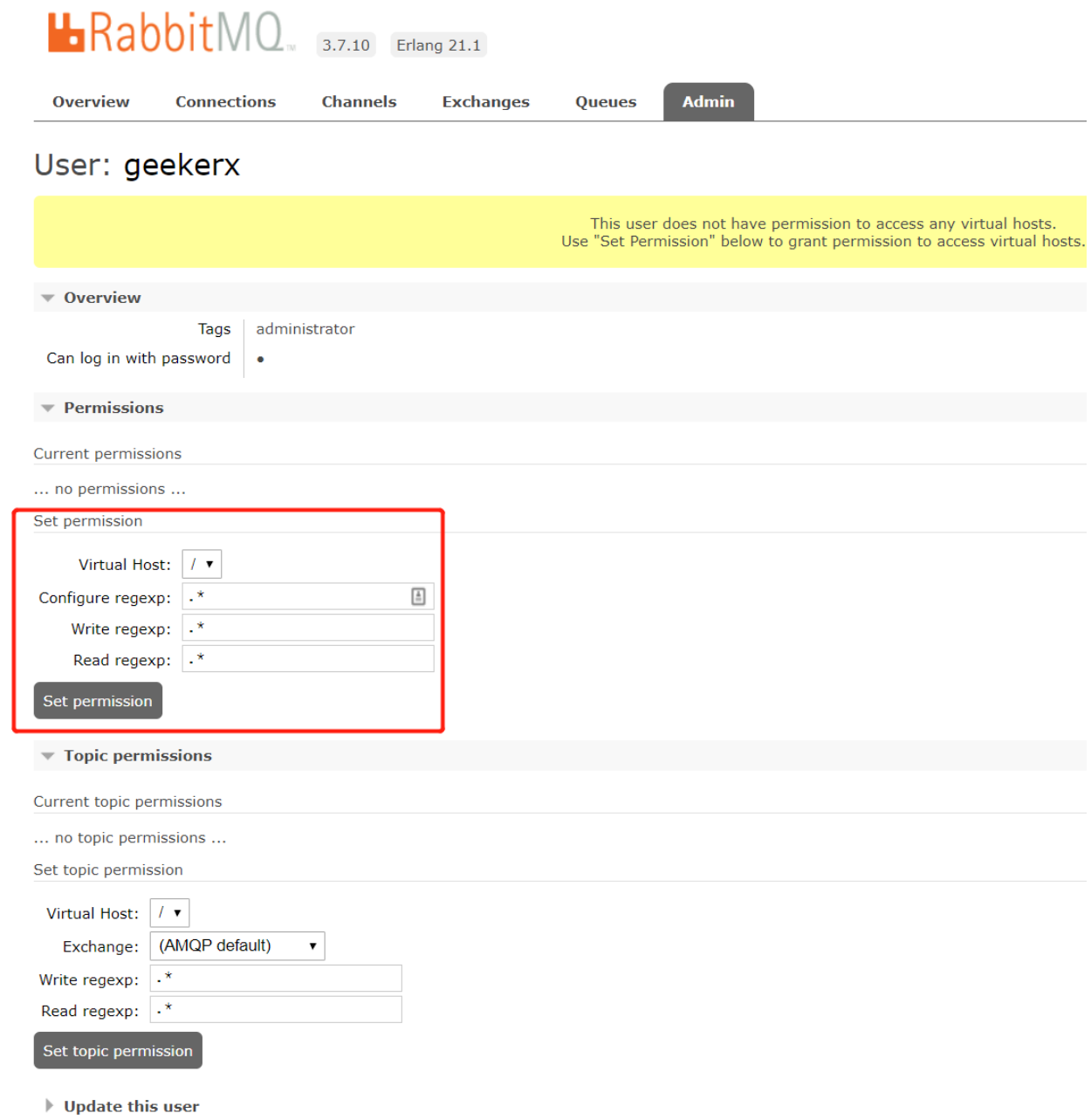
看到rabbitMQ管理界面，geekerx用户没有虚拟路径的权限

> 参考：
> https://blog.csdn.net/luwei42768/article/details/53730960

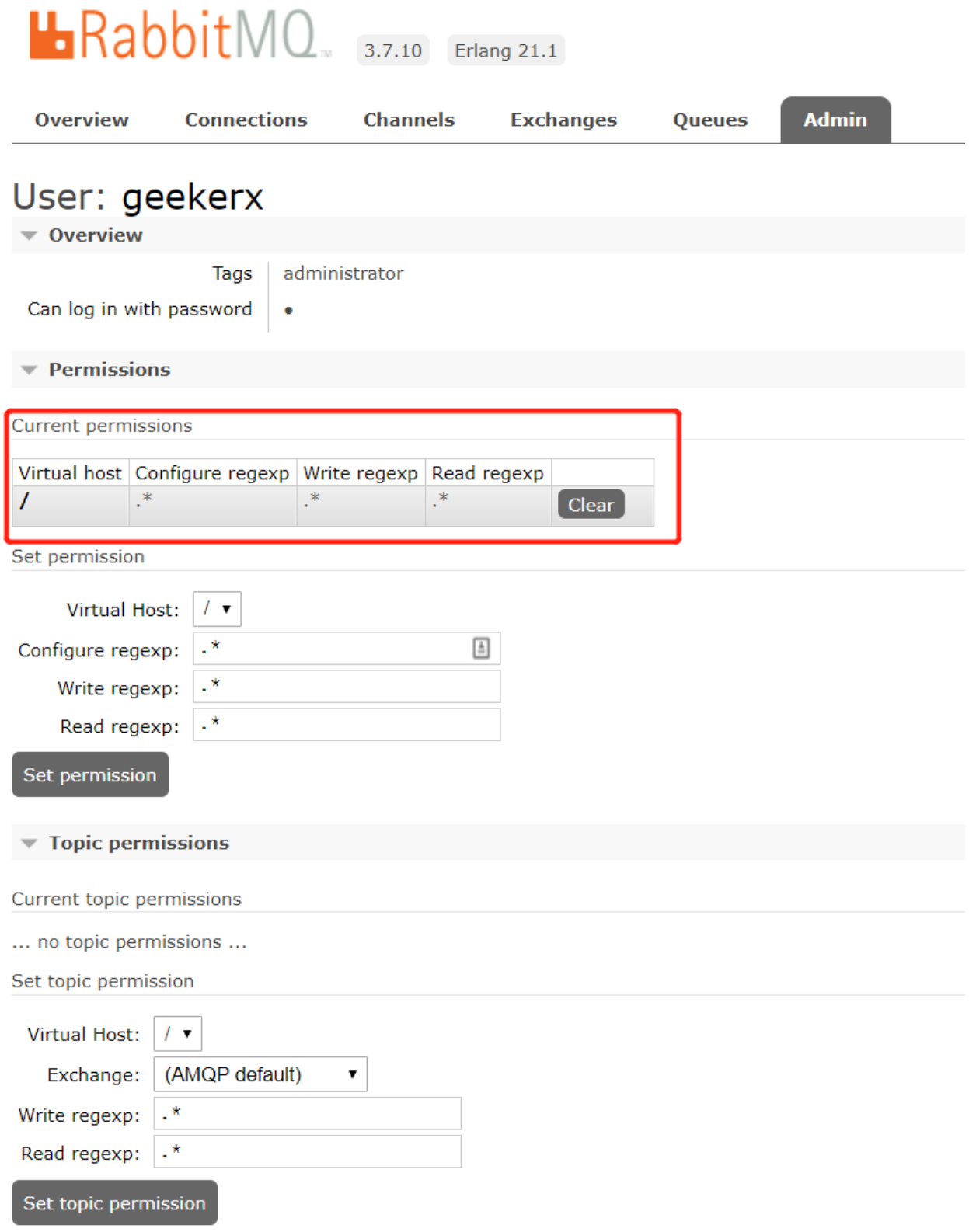



点击geekerx用户名，进入用户设置界面

直接根据默认配置，点击"Set permission"




保证Current permissions下面有值即可



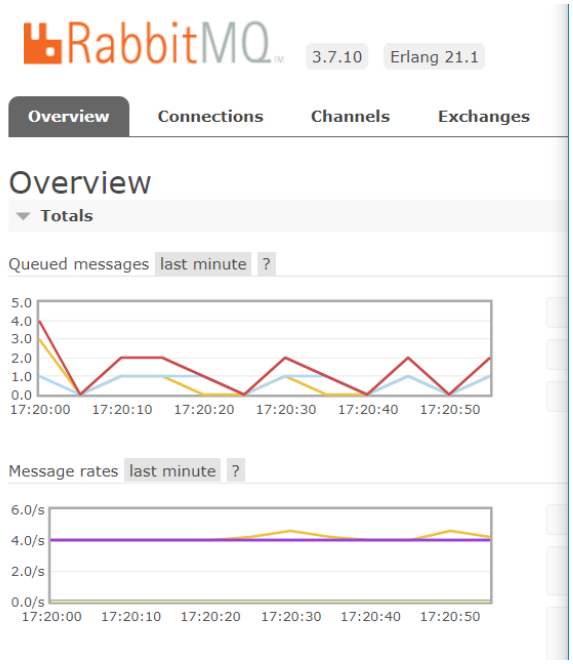

设置完用户权限后，再次启动则启动成功！

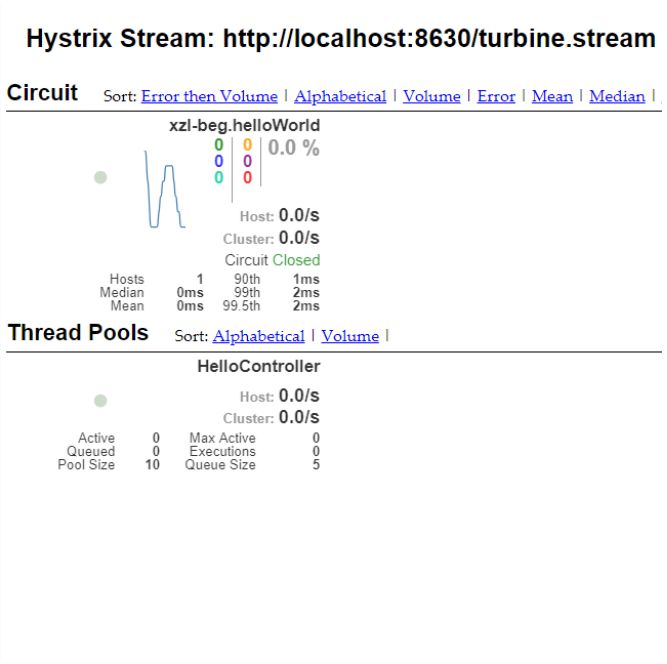
接下来运行时，turbine会报错**（但是没有复现）**：

```
2019-01-23 16:51:49.383 ERROR 20828 --- [ctor-http-nio-3] o.s.w.s.adapter.HttpWebHandlerAdapter : Unhandled failure: 你的主机中的软件中止了一个已建立的连接。, response already set (status=null)
```

```
2 2019-01-23 16:51:49.384 WARN 20828 --- [ctor-http-nio-3] o.s.h.s.r.ReactorHttpHandlerAdapter : Handling completed with error: 你的主机中的软件中止了一个已建立的连接。
```

多次请求 http://localhost:8700/hello 后，rabbitMQ和dashboard监控情况如下




beg服务日志：

```
1 2019-01-23 16:59:57.097 INFO [zipkin-beg,,,] 15144 --- [ask-scheduler-2] o.s.a.r.c.CachingConnectionFactory : Attempting to connect to: [192.168.100.150:5672]
2 2019-01-23 16:59:57.102 INFO [zipkin-beg,,,] 15144 --- [ask-scheduler-2] o.s.a.r.c.CachingConnectionFactory : Created new connection: rabbitConnectionFactory.publisher#23240e35:0/SimpleConnection@64151b26 [delegate=amqp://geekerx@192.168.100.150:5672/, localPort= 51927]
```

# 八 zuul代理其他微服务

## 1.配置

### ①maven依赖

```
1 <dependency>
2  <groupId>org.springframework.boot</groupId>
3  <artifactId>spring-boot-starter-actuator</artifactId>
4 </dependency>
5 <dependency>
6  <groupId>org.springframework.cloud</groupId>
```

```
  <artifactId>spring-cloud-starter-consul-discovery</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-zuul</artifactId>
</dependency>
```

**②zuul相关配置**

```
zuul:
 # 忽略框架默认的服务映射路径
 ignored-services: '*'
 # 不忽略框架与权限相关的头信息
 ignore-security-headers: false
 # 转发后请求的服务是否还带上转发表示字符
 strip-prefix: false
# 如果路由是通过URL指定的，那么需要配置zuul.host.connect-timeout-millis和zuul.host.socket-timeout-millis
# host:
 # connect-timeout-millis: 20000
 # socket-timeout-millis: 20000
  # 不忽略任何头部信息，所有header都转发到下游的资源服务器
  # 所以这里对制定的路由开启自定义敏感头。除了设置为true，也可以设置为空。zuul.sensitiveHeaders= 只是全局设置的做法。不推荐！破坏了默认设置的用意
 # sensitive-headers:
  routes:
  # 所有以/config开头的请求都转发到xzl-config应用中
  xzl-config:
  path: /config/**
  serviceId: xzl-config
  sensitiveHeaders: true
  xzl-turbine:
  path: /turbine/**
  serviceId: xzl-turbine
  xzl-hystrix-dashboard:
  path: /dashboard/**
  serviceId: xzl-hystrix-dashboard
  xzl-beg:
  path: /beg/**
  serviceId: xzl-beg
  # springcloud项目中经过网关zuul转发请求后发生session失效问题，这是由于zuul默认会丢弃原来的session并生成新的session
  sensitiveHeaders: true
```

其中几个参数强调解释一下：

**①zuul.ignore-security-headers=false**

不忽略框架与权限相关的头信息，保证各服务在代理后头部保存的session、token等信息完整。

**②zuul.strip-prefix=false**

转发后请求的服务是否还带上转发表示字符。

具体如下，原始的url是https://zuulIp:zuulPort/demo/hello,

如果这个参数设置为true，那么对于Demo1的服务而言,它受到的请求url就是https://demoIp:demoPort/hello

如果这个参数设置为false，那么对于Demo1的服务而言,它受到的请求url就是https://demoIp:demoPort/demo/hello

**③zuul.routes.xzl-config.sensitiveHeaders=true**

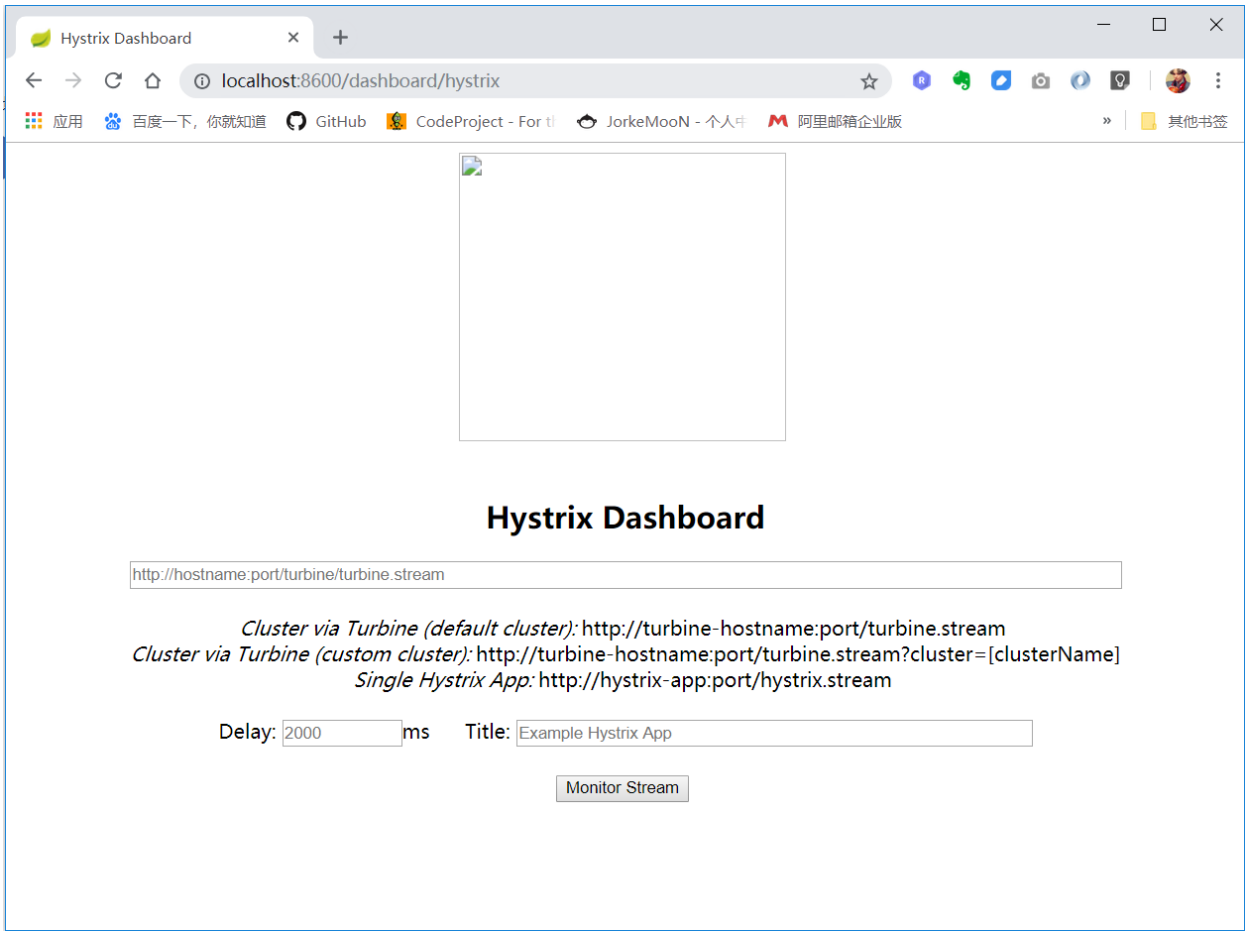
不忽略任何头部信息，所有header都转发到下游的资源服务器。所以这里对制定的路由开启自定义敏感头。除了设置为true，也可以设置为空。

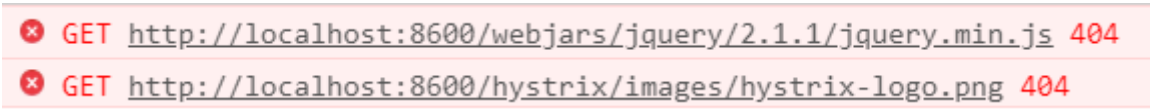
zuul.sensitiveHeaders= 只是全局设置的做法。不推荐！破坏了默认设置的用意。

## 2.静态资源问题

zuul代理其他微服务时出现了一个问题：

代理其他微服务（beg、config、turbine）都没问题，但是代理hystrix dashboard会出现下图静态资源丢失，且[Monitor Stream]按钮无效的问题




LOGO静态图片原路径为：http://localhost:8630/hystrix/images/hystrix-logo.png

代理后路径为：http://localhost:8600/hystrix/images/hystrix-logo.png

仅仅更换了端口号，代理后的地址是没资源的

代理后正确的资源路径应该为：

http://localhost:8600/dashboard/hystrix/images/hystrix-logo.png

由此，可以引申出zuul对于代理静态资源的一些弊端，另起一篇

-> Zuul VS Nginx.note

所以，这里不再使用zuul代理hystrix dashboard这类包含静态资源的微服务。

## 3.跨域问题

正常情况下，跨域是这样的：

1. 微服务配置跨域+zuul不配置=有跨域问题

2. 微服务配置+zuul配置=有跨域问题
3. 微服务不配置+zuul不配置=有跨域问题
4. 微服务不配置+zuul配置=ok

然而每个服务自己有跨域解决方案，而网关需要做最外层的跨域解决方案。如果服务已有跨域配置网关也有，会出现*多次配置问题。

```
Access-Control-Allow-Origin:"*,*"
也就是multiple Access-Control-Allow-Origin
```

> 参考：
> https://blog.csdn.net/moshowgame/article/details/80507696

在测试环境部署的zuul，在本地启动的beg跪求业务项目，通过zuul访问代理的beg服务会报500错误

```
{
 "timestamp": "2019-01-24T09:12:51.997+0000",
 "status": 500,
 "error": "Internal Server Error",
 "message": "GENERAL"
}
```

将beg服务部署到zuul同一个主机上就没有问题

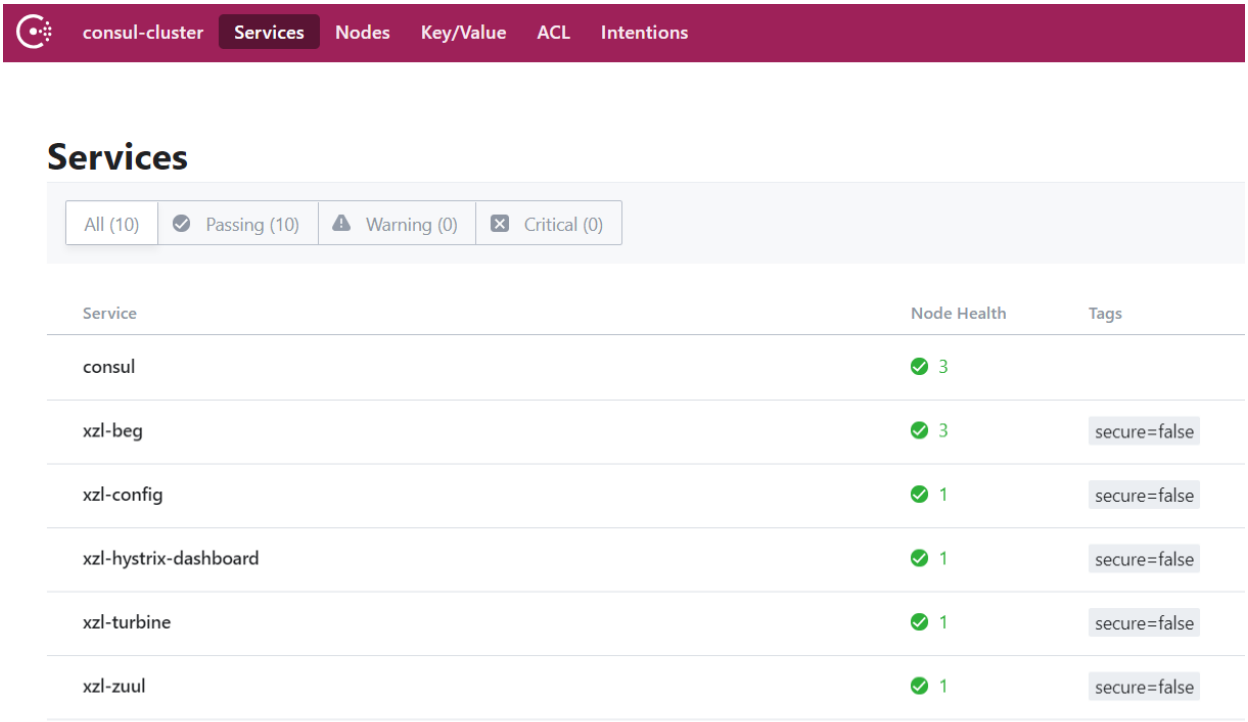
## 这是哪块的跨域问题呢?

# 九 部署到测试环境

整体都配置完毕后，将所有服务都打成jar包，部署到测试环境中

打jar包注意事项移步 -> 打Jar包部署细节.note

微服务部署情况如下：




这里xzl-beg跪求更换端口部署3个微服务，组建成xzl-beg集群服务

consul注册服务和zuul网关服务会自动实现负载均衡

请求xzl-beg服务默认会实现**轮询机制**

示例：

连续请求三次 http://192.168.100.150:8700/beg/hello

返回结果分别为：

```
Hello World!
端口号：8810
```

```
Hello World!
端口号：8811
```

```
Hello World!
端口号：8812
```