系统稳定性——Dubbo 常见错误及解决方法

> 作者：空冥
> 创作日期：2019-07-18
> 专栏地址：【稳定大于一切】

Dubbo作为高性能 RPC（Remote Procedure Call）框架已经成为 Apache 的顶级项目，意味着在全球被数以千计的公司所采用来其实现其分布式架构的互联集成，尤其是在国内更受欢迎。下面根据我们自身遇到的问题，加上用户提供的一些反馈，来大致梳理下 Dubbo 的常见错误及解决方法。

目录

1. 地址找不到：No provider available

找不到服务，这时候可能有这么几种情况：

- Provider 服务没启动，或者注册中心（比如 ZooKeeper，Nacos，Consul）宕机了。
- Dubbo 的服务配置有误差，必须保证服务名，组别(默认是 Dubbo)，version 三者都正确。
- 访问的环境有误：通常我们会有开发环境、测试环境、线上生产环境等多套环境。有时候发布的服务到了测试环境，而访问调用时却走了开发环境。

排查步骤

1. 访问注册中心的 Ops 系统，查询对应的服务是否有提供者列表；同时检查调用者应用所在服务器的日志(一般每种注册服务的客户端都会有对应的日志记录)，查看是否有地址信息的推送/拉取记录。
2. 如无，则表明发布者发布服务失败，检查发布者的应用启动是否成功。
3. 如有服务，则检查调用者应用所连接的注册中心，确认跟预期的环境要匹配。
4. 如上述都没有问题，检查是否配置了路由过滤的规则等。

2. 调用超时：client-side timeout

一般超时是调用端发生在请求发出后，无法在指定的时间内获得对应的响应。原因大概有以下几种情况：

- 服务端确实处理比较慢，无法在指定的时间返回结果，调用端就自动返回一个超时的异常响应来结束此次调用。
- 服务端如果响应的比较快，但当客户端 Load 很高，负载压力很大的时候，会因为客户端请求发不出去、响应卡在 TCP Buffer 等问题，造成超时。因为客户端接收到服务端发来的数据或者请求服务端的数据，都会在系统层面排队，如果系统负载比较高，在内核态的时间占比就会加长，从而造成客户端获取到值时已经超时。
- 通常是业务处理太慢，可在服务提供方机器上执行：jstack [PID] > jstack.log 分析线程都卡在哪个方法调用上，这里就是慢的原因。如果不能调优性能，请调高 timeout 阈值。

排查和解决步骤

1. 两边可能有GC,检查服务端和客户端 GC 日志，耗时很长的 GC，会导致超时。超时的发生很可能意味着调用端或者服务端的资源(CPU，内存或者网络)出现了瓶颈，需要检查服务端的问题还是调用端的问题来排除GC抖动等嫌疑。
2. 检查服务端的网络质量，比如重传率来排除网络嫌疑。
3. 借助链路跟踪的分析服务（比如阿里的 ARMS，开源的 OpenTracing 系的实现 Zipkin、SkyWalking 等）来分析下各个点的耗时情况。

3. 服务端的线程资源耗尽：Thread pool is EXHAUSTED

Dubbo服务端的业务线程数是 200 个，如果多个并发请求量超过了 200，就会拒绝新的请求，抛出此错误。这种问题有这么几种解决办法：

排查和解决步骤

- 调整 Provider 端的 dubbo.provider.threads 参数的大小，调大一些即可。
- 调整 Consumer 端的 dubbo.consumer.actives 参数的大小，调小一些即可。
- 增加 Provider 服务的数量，分担压力。

4. Hessian 序列化失败：HessianRuntimeException

- 检查服务方法的传入传出参数是否实现 Serializable 接口。
- 检查服务方法的传入传出参数是否继承了 Number、Date、ArrayList、HashMap 等 Hessian 特殊化处理的类。

5. 启动时 Configuration problem: Unable to locate Spring NamespaceHandler for XML schema

表示 Spring 找不到<dubbo:...>配置的解析处理器。通常是 Dubbo 的 jar 包没有被引入，请添加对 Dubbo 的依赖；或者是 ClassLoader 隔离，查看是否有使用 OSGI 或其它热加载机制。

推荐链接

- Dubbo on GitHub

加入我们

【稳定大于一切】打造国内稳定性领域知识库，**让无法解决的问题少一点，让世界的确定性多一点。**

- GitHub 地址
- 钉钉群号：23179349