1、面试题

如何基于dubbo进行服务治理、服务降级、失败重试以及超时重试？

2、面试官心里分析

服务治理，这个问题如果问你，其实就是看看你有没有服务治理的思想，因为这个是做过复杂微服务的人肯定会遇到的一个问题。

服务降级，这个是涉及到复杂分布式系统中必备的一个话题，因为分布式系统互相来回调用，任何一个系统故障了，你不降级，直接就全盘崩溃？那就太坑爹了吧

失败重试，分布式系统中网络请求如此频繁，要是因为网络问题不小心失败了一次，是不是要重试？

超时重试，同上，如果不小心网络慢一点，超时了，如何重试？

（1）dubbo工作原理：服务注册，注册中心，消费者，代理通信，负载均衡

（2）网络通信、序列化：dubbo协议，长连接，NIO，hessian序列化协议

（3）负载均衡策略，集群容错策略，动态代理策略：dubbo跑起来的时候一些功能是如何运转的，怎么做负载均衡？怎么做集群容错？怎么生成动态代理？

（4）dubbo SPI机制：你了解不了解dubbo的SPI机制？如何基于SPI机制对dubbo进行扩展？

（5）dubbo的服务治理、降级、重试

3、面试题剖析

（1）服务治理

1）调用链路自动生成

一个大型的分布式系统，或者说是用现在流行的微服务架构来说吧，分布式系统由大量的服务组成。那么这些服务之间互相是如何调用的？调用链路是啥？说实话，几乎到后面没人搞的清楚了，因为服务实在太多了，可能几百个甚至几千个服务。

那就需要基于dubbo做的分布式系统中，对各个服务之间的调用自动记录下来，然后自动将各个服务之间的依赖关系和调用链路生成出来，做成一张图，显示出来，大家才可以看到对吧。

服务A -> 服务B -> 服务C

-> 服务E

-> 服务D

-> 服务F

-> 服务W

2）服务访问压力以及时长统计

需要自动统计各个接口和服务之间的调用次数以及访问延时，而且要分成两个级别。一个级别是接口粒度，就是每个服务的每个接口每天被调用多少次，TP50，TP90，TP99，三个档次的请求延时分别是多少；第二个级别是从源头入口开始，一个完整的请求链路经过几十个服务之后，完成一次请求，每天全链路走多少次，全链路请求延时的TP50，TP90，TP99，分别是多少。

这些东西都搞定了之后，后面才可以来看当前系统的压力主要在哪里，如何来扩容和优化啊

3）其他的

服务分层（避免循环依赖），调用链路失败监控和报警，服务鉴权，每个服务的可用性的监控（接口调用成功率？几个9？）99.99%，99.9%，99%

（2）服务降级

比如说服务A调用服务B，结果服务B挂掉了，服务A重试几次调用服务B，还是不行，直接降级，走一个备用的逻辑，给用户返回响应

public interface HelloService {

void sayHello();

}

public class HelloServiceImpl implements HelloService {

public void sayHello() {

System.out.println("hello world......");

}

}

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd http://code.alibabatech.com/schema/dubbo http://code.alibabatech.com/schema/dubbo/dubbo.xsd">

<dubbo:application name="dubbo-provider" />

<dubbo:registry address="zookeeper://127.0.0.1:2181" />

<dubbo:protocol name="dubbo" port="20880" />

<dubbo:service interface="com.zhss.service.HelloService" ref="helloServiceImpl" timeout="10000" />

<bean id="helloServiceImpl" class="com.zhss.service.HelloServiceImpl" />

</beans>

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd http://code.alibabatech.com/schema/dubbo http://code.alibabatech.com/schema/dubbo/dubbo.xsd">

<dubbo:application name="dubbo-consumer" />

<dubbo:registry address="zookeeper://127.0.0.1:2181" />

<dubbo:reference id="fooService" interface="com.test.service.FooService" timeout="10000" check="false" mock="return null">

</dubbo:reference>

</beans>

现在就是mock，如果调用失败统一返回null

但是可以将mock修改为true，然后在跟接口同一个路径下实现一个Mock类，命名规则是接口名称加Mock后缀。然后在Mock类里实现自己的降级逻辑。

public class HelloServiceMock implements HelloService {

public void sayHello() {

// 降级逻辑

}

}

（3）失败重试和超时重试

所谓失败重试，就是consumer调用provider要是失败了，比如抛异常了，此时应该是可以重试的，或者调用超时了也可以重试。

<dubbo:reference id="xxxx" interface="xx" check="true" async="false" retries="3" timeout="2000"/>

某个服务的接口，要耗费5s，你这边不能干等着，你这边配置了timeout之后，我等待2s，还没返回，我直接就撤了，不能干等你

如果是超时了，timeout就会设置超时时间；如果是调用失败了自动就会重试指定的次数

你就结合你们公司的具体的场景来说说你是怎么设置这些参数的，timeout，一般设置为200ms，我们认为不能超过200ms还没返回

retries，3次，设置retries，还一般是在读请求的时候，比如你要查询个数据，你可以设置个retries，如果第一次没读到，报错，重试指定的次数，尝试再次读取2次