redis响应变慢，查看日志，发现大量TimeoutException。

大量TimeoutException，说明当前redis服务节点上已经堆积了大量的连接查询，超出redis服务能力，再次尝试连接的客户端，redis 服务节点直接拒绝，抛出错误。

那到底是什么导致了这种情况的发生呢？

总结起来，我们可以从以下几方面进行关注：

### 1、redis服务节点受到外部关联影响

redis服务所在服务器，物理机的资源竞争及网络状况等。同一台服务器上的服务必然面对着服务资源的竞争，CPU，内存，固存等。

#### 1.1 CPU资源竞争

redis属于CPU密集型服务，对CPU资源依赖尤为紧密，当所在服务器存在其它CPU密集型应用时，必然会影响redis的服务能力，尤其是在其它服务对CPU资源消耗不稳定的情况下。

因此，在实际规划redis这种基础性数据服务时应该注意一下几点：

* 一般不要和其它类型的服务进行混部。
* 同类型的redis服务，也应该针对所服务的不同上层应用进行资源隔离。

说到CPU关联性，可能有人会问是否应该对redis服务进行CPU绑定，以降低由CPU上下文切换带来的性能消耗及关联影响？

简单来说，是可以的，这种优化可以针对任何CPU亲和性要求比较高的服务，但是在此处，有一点我们也应该特别注意：我们在 关于redis内存分析，内存优化 中介绍内存时，曾经提到过子进程内存消耗，也就是redis持久化时会fork出子进程进行AOF/RDB持久化任务。

对于开启了持久化配置的redis服务（一般情况下都会开启），假如我们做了CPU亲和性处理，那么redis fork出的子进程则会和父进程共享同一个CPU资源，我们知道，redis持久化进程是一个非常耗资源的过程，这种自竞争必然会引发redis服务的极大不稳定。

#### 1.2 内存不在内存了

关于redis内存分析，内存优化 开篇就讲过，redis最重要的东西，内存。

内存稳定性是redis提供稳定，低延迟服务的最基本的要求。

然而，我们也知道操作系统有一个 swap 的东西，也就将内存交换到硬盘。假如发生了redis内存被交换到硬盘的情景发生，那么必然，redis服务能力会骤然下降。

swap发现及避免：

1）info memory：

关于redis内存分析，内存优化 中我们也讲过，swap这种情景，此时，查看redis的内存信息，可以观察到碎片率会小于1。这也可以作为监控redis服务稳定性的一个指标。

2）通过redis进程查看

首先通过 info server 获取进程id：



查看redis进程swap情况：cat/proc/9007/smaps



确定交换量都为0KB或者4KB。

3）redis服务maxmemory配置

关于redis内存分析，内存优化 中我们提到过，对redis服务必要的内存上限配置，这是内存隔离的一种必要。需要确定的是所有redis实例的分配内存总额小于总的可用物理内存。

4）系统优化：

另外，在最初的基础服务操作系统安装部署时，也需要做一些必要的前置优化，如关闭swap或配置系统尽量避免使用

#### 1.3 网络问题

1）网络资源耗尽

简单来说，就是带宽不够了，整个属于基础资源架构的问题了，对网络资源的预估不足，跨机房，异地部署等都会成为诱因。

2）连接数用完了

一个客户端连接对应着一个TCP连接，一个TCP连接在LINUX系统内对应着一个文件句柄，系统级别连接句柄用完了，也就无法再进行连接了。（更多面试题，欢迎关注公众号 Java面试题精选）

查看当前系统限制：ulimit -n

设置：ulimit -n {num}

3）端口TCP backlog队列满了

linux系统对于每个端口使用backlog保存每一个TCP连接。

redis配置：tcp\_backlog 默认511



高并发情境下，可以适当调整此配置，但需要注意的是，同时要调整系统相关设置。

系统修改命令：echo {num}>/proc/sys/net/core/somaxconn

查看因为队列溢出导致的连接绝句：netstat -s | grep overflowed



4）网络延迟

网络质量问题，可以使用 redis-cli 进行网络状况的测试：

延迟测试：redis-cli -h {host} -p {port} --latency



采样延迟测试：redis-cli -h {host} -p {port} --latency-history 默认15s一次



图形线上测试结果：redis-cli -h {host} -p {port} --latency-dist



4）网卡软中断

单个网卡队列只能使用单个CPU资源问题。

### 2、redis 服务使用问题

#### 2.1 慢查询

如果你的查询总是慢查询，那么必然你的使用存在不合理。

1）你的key规划是否合理

太长或太短都是不建议的，key需要设置的简短而有意义。

2）值类型选择是否合理。

hash还是string，set还是zset，避免大对象存储。

线上可以通过scan命令进行大对象发现治理。

3）是否能够批查询

get还是mget；是否应该使用pipeline。

4）禁止线上大数据量操作

#### 2.2 redis服务运行状况

查看redis服务运行状况：redis-cli -h {host} -p {port} --stat



keys：当前key总数；mem：内存使用；clients：当前连接client数；blocked：阻塞数；requests：累计请求数；connections：累计连接数

#### 2.3 持久化操作影响

1）fork子进程影响

redis 进行持久化操作需要fork出子进程。fork子进程本身如果时间过长，则会产生一定的影响。

查看命令最近一次fork耗时：info stats



单位微妙，确保不要超过1s。

2）AOF刷盘阻塞

AOF持久化开启，后台每秒进行AOF文件刷盘操作，系统fsync操作将AOF文件同步到硬盘，如果主线程发现距离上一次成功fsync超过2s，则会阻塞后台线程等待fsync完成以保障数据安全性。

3）THP问题

关于redis内存分析，内存优化 中我们讲过透明大页问题，linux系统的写时复制机制会使得每次写操作引起的页复制由4KB提升至2M从而导致写慢查询。如果慢查询堆积必然导致后续连接问题。