Ghost:20:33:27

这个根节点是啥子？？？ leader？

答: 不是.leader 只是 zk 服务的一个状态.根节点是 zk 服务的存储结构.

黄欣健:20:34:14

根节点是每个 server 下的存储结构吗

答: 对的.每个 Server 下都有一个.

黄欣健:20:35:44

这些节点是根据 ZXID 排序的吗

答: 它们之间没有排序的.

Free3610:20:38:23

b 树吗

答: 不是.

面试题：
1）一个客户端到底连接的是 zk 列表中的哪台 Server？
答：在客户端获取到 zk 列表后，首先会将 zk 列表打散，然后对打散后的列表采取轮询的方式进行连接尝试，直到连接成功。
2）第一个客户端将 zk 列表打散后，在打散的列表上采取轮询的方式尝试连接，那么，第二个客户端又来连接 zk 集群，其是在前面打散的基础之上采用轮询方式选择 Server，还是又重新打散后再进行轮询连接尝试？
答：又进行打散，然后再进行轮询连接尝试。

攀登:20:49:39

负载均衡？

攀登:20:50:32

打乱你随机

攀登:20:50:59

貌似还有很多种，

黄欣健:20:52:57

继续打散

攀登:20:52:57

轮询、随机、响应速度，最小连接数，处理能力，DNS 响应

杨腾飞:20:54:51

炒鸡蛋

杨腾飞:20:55:31

俩鸡蛋不够，再加一个鸡蛋。肯定继续打散

溼汐:21:02:30

打乱

吕伟明:21:02:33

乱序

攀登:21:08:25

一旦有连接成功了美酒返回了吧

答: 一旦连接成功了立马就把结果返回了

攀登:21:10:55

n 个节点，全部尝试 3*n 次，全部失败就返回异常

攀登:21:11:29

全部轮询了一遍

攀登:21:12:28

延时

攀登:21:13:03

上一次轮询的第一个索引

黄欣健:21:13:12

上一次的结尾

跑丢一只鞋-北京-2 年:21:13:42

diyigif

Ghost:21:13:55

一个 server？》

答: 对的,只有一台 zk 的时候会出现这样的情况.

跑丢一只鞋-北京-2 年:21:13:59

第一个 if

大胜.彼得:21:15:31

每次都打散再轮训，是为了尽可能的平均分配连接，防止连接分配不均匀。对吗？

答: 说的很正确.

湮汐:21:16:02

防止多个客户端

黄欣健:21:16:07

就是负载吧

Ghost:21:16:31

感觉是不加权的那种均衡算法

答: 对的,就是不加权的.

攀登:21:16:43

随机轮询均衡

攀登:21:18:07

客户端每一次连接都先随机，再轮询。

答: 是的.

马崇-java-上海-1 年:21:19:31

理解了，

马崇-java-上海-1 年:21:19:36

但是代码没理解

张中强:21:19:38

这种算法好吗

马崇-java-上海-1 年:21:19:39

业务理解了

Ghost:21:20:26

理解是理解了 但是我们为啥不先说 zk 的典型使用 我这种没用过的渣渣 搞了半天不知道为啥要连他

答: 这是内容的先后顺序问题,现在讲的是为后面的内容做铺垫,不然直接讲典型使用会听不懂.

跑丢一只鞋-北京-2 年:21:20:32

如果一个客户端 第一次 连接失败，等待一秒后 会接着 尝试连接 第一次 连接失败的主机吗？

答: 比如三台 zk,客户端第一次连接之前先打乱顺序,比如打乱后的顺序是 312,现在开始连接 3,失败了就连接 1,再失败连接 2,如果还失败就休息一秒后继续连接 3,是这么一个顺序.

默认尝试连接次数可以自己设置,如果是 3,也就是可以走三遍 312 的流程.

没有这首歌。:21:21:28

最近要用 kafka 搭建消息平台。发现 kafka 里面自带了 zk

答: 对的,kafka 自带 zk,但是我们一般都用外部的.

攀登:21:22:33

轮询，轮询 3 轮，三轮过后还没成功，就返回错误

今天不傻:21:23:16

现在最大的困惑是，听好像听会了些，但完全不会用

答: 现在不要着急,等到了 zk 典型应用场景那一块还有讲到 Dubbo 和 Kafka 的时候,有的是机会用.

黄欣健:21:23:38

现在就是打 zk 的地基，后面讲应用应该会串起来

答: 是的,健哥分析的很到位.

湮汐:21:23:45

这个策略的代码没看懂。。。

石将从:21:23:59

你看后面的 dubbo 录播就看到用法的

谢尊锦:21:24:21

zk 应用很简单的

石将从:21:24:26

做注册中心

黄欣健:21:25:04

雷哥 ZK 精华

马崇-java-上海-1 年:21:25:42

那我嗝屁了

谢尊锦:21:25:51

比源码简单

攀登:21:26:28

高可靠、高可用

黄欣健:21:27:13

简历多加一行：精通 Zookeeper

马崇-java-上海-1 年:21:27:35

建哥牛逼

黄欣健:21:27:59

我不精通，我说你们这些大佬

攀登:21:28:40

超时呗

大胜.彼得:22:02:35

下面绿色的宽度也是 5t 吗

答: 不是

黄欣健:22:03:17

每次心跳后，都会换桶重新计算

答: 每次心跳都会重新计算,但是不一定会换桶.

攀登:22:03:20

根据心跳更新 sessionid 所在的桶，桶超时，移除桶内的 sessionID

答: 理解的很到位.

张庆辉:22:03:28

最长时间 是 会话超时时间+5t

答: 不一定.

马崇-java-上海-1 年:22:04:55

如果有连接的话，会一直发心跳吗

答: 如果有连接就会一直发心跳,如果没发心跳,就认为你挂了.

杨腾飞:22:06:51

超时时间恰好是桶呢边界

答: 这个不影响,就是判断小于还是小于等于的问题

马崇-java-上海-1 年:22:07:07

意思是桶里会话就是表示超时了，就没有发心跳了吗

答: 对的,你没发心跳,所以才超时.

马崇-java-上海-1 年:22:08:05

这个桶是哪里来的呢，有多少桶啊

湮汐:22:08:20

感觉这个算法 比不上 LRU 啊

答: 肯定比 LRU 好

黄欣健:22:08:22

桶是动态创的吗

答: 对的.

攀登:22:08:44

超时移除 sessionID,有延时，延时中又来心跳，是会话转移吗

大胜.彼得:22:23:39

心跳一来 就执行到这段程序是吧

答: 不是,这里只是刚创建会话.

张中强:22:23:42

为什么我延迟

任献良:22:26:24

说明没过期

马崇-java-上海-1 年:22:26:34

还没过期吧

跑丢一只鞋-北京-2 年:22:26:35

还没有走到 超时的桶

张中强:22:26:38

没失效

湮汐:22:26:39

不需要改到新的桶

黄海-成都:22:26:41

需要还桶

攀登:22:26:43

要换桶

黄欣健:22:27:31

相当于上面的那个小箭头吗

湮汐:22:30:31

重新校验？

吕伟明:22:30:32

验证

吕伟明:22:30:37

重新

大胜.彼得:22:30:38

重验

张中强:22:31:33

是的

马崇-java-上海-1 年:22:38:24

会话有效

任献良:22:39:26

不

跑丢一只鞋-北京-2 年:22:39:34

不超时啊

马崇-java-上海-1 年:22:39:37

不超时

湮汐:22:39:37

超

攀登:22:39:43

不超

攀登:22:39:54

现实的时间线还没到 15t

马德成:22:41:13

那会话怎么会跑到 15 呢？

大胜.彼得:22:41:38

s.ticktim 这个值 是怎么来的

吕伟明:22:42:29

不超时

攀登:22:42:30

不超

溼汐:22:42:32

不超

马德成:22:42:37

不超时

攀登:22:43:07

要换桶例如

黄欣健:22:43:21

有点懵

攀登:22:44:02

超时时间跨桶了，需要换桶

马德成:22:45:35

换桶的问题明白，但是超时不超时很懵

大胜.彼得:22:45:45

上一个心跳的超时桶 小于下一个心跳超时桶 就要换桶

湮汐:22:47:18

s.tickTime 表示当前属于哪个桶，expireTime 是本次心跳来了应该放到哪个桶

攀登:22:48:06

对

马德成:22:49:39

刚才那块不是说只是换桶的事么？怎么又出现超市的注解了？懵了。。。

马德成:22:50:54

s.tickTime >= expireTime 是需要换桶，但是怎么判断是超时了呢？

答:如果 s.tickTime >= expireTime,则表示当前会话所在的桶和当前心跳所在的桶在同一个桶里,这时候什么都不用做,也不需要换桶.

攀登:22:52:53

未来的桶的超时时间

黄欣健:22:54:28

s.tickTime 是这个 Session 所在的桶，expireTime 是新来的

心跳所在的桶，是这样理解吗

答: 是的

任献良:22:54:43

是的

攀登:22:54:45

会话所在的桶没有超时，即时间没到

大胜.彼得:22:55:02

是上一次数据所在桶

任献良:22:55:04

不是 键哥，

攀登:22:55:42

湮汐说了：

s.tickTime 表示当前属于哪个桶，expireTime 是本次心跳来了应该放到哪个桶

马德成:22:56:03

是当前会话的超时间所在桶与上次的桶相同，所以不用换桶

答: 对的

跑丢一只鞋-北京-2 年:22:56:25

没换桶 肯定不超时。 换了桶也不一定超时，只要 到了桶的时间边缘 还没有 当前会话还没有心跳来，那就超时了，

超时了 就删除 sessionId

答: 是的

黄欣健:22:56:59

小懵，得看看录播

马德成:22:57:14

换桶是清楚的

马崇-java-上海-1 年:22:57:14

是到桶的边缘才会超时吧

答: 是的

攀登:22:57:20

到清除超时桶

马德成:22:57:23

超时是不清楚的

攀登:22:57:47

上面的 run 方法

马崇-java-上海-1 年:22:57:47

心跳在当前桶，然后会话到了当前边缘了，就会超时

马德成:22:58:42

所以超时是会话时间点与桶边缘的比较？

吕伟明:22:59:39

这个搜索快捷键是啥

攀登:22:59:55

两次 shift

马德成:23:00:22

记录当前桶的时间

攀登:23:00:24

下一次的桶超时间

攀登:23:00:48

就是当前时间线所在的会话桶

攀登:23:02:42

5t

马德成:23:02:42

5t

余爽:23:02:45

5t

大胜.彼得:23:02:48

5t

吕伟明:23:02:55

5t

大胜.彼得:23:03:10

没

没有这首歌。:23:03:13

.没有

马德成:23:03:13

没超市

攀登:23:03:16

没呀

马德成:23:07:10

所以是否超时的判断是以收到的当前心跳时间 与 桶边缘的对比？

马德成:23:08:02

理解了