**一、本文目的**

       演示在一台机器上搭建3主3从的redis集群，通过演示了解redis集群的搭建，使用和注意事项

**二、搭建说明**

       1、同一台机器搭建3主3从的伪集群

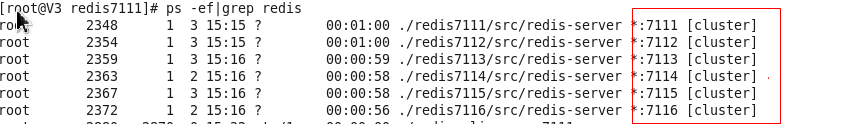
       2、使用端口7111~7116，集群端口为端口前+1,如果使用端口7111，则其集群通信端口为17111（集群内部通信端口，自动设置）,

             redis.config的配置如下：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | daemonize | port | pidfile | logfile | cluster-enabled | cluster-config-file | cluster-node-timeout |
| 7111 | yes | 7111 | /var/run/redis7111.pid | "redis7111.log" | yes | nodes-7111.conf | 5000 |
| 7112 | yes | 7112 | /var/run/redis7112.pid | "redis7112.log" | yes | nodes-7112.conf | 5000 |
| 7113 | yes | 7113 | /var/run/redis7113.pid | "redis7113.log" | yes | nodes-7113.conf | 5000 |
| 7114 | yes | 7114 | /var/run/redis7114.pid | "redis7114.log" | yes | nodes-7114.conf | 5000 |
| 7115 | yes | 7115 | /var/run/redis7115.pid | "redis7115.log" | yes | nodes-7115.conf | 5000 |
| 7115 | yes | 7116 | /var/run/redis7116.pid | "redis7116.log" | yes | nodes-7116.conf | 5000 |

           3、需要打开7111~7116、17111~17116一共12个端口

**三、搭建过程**

         1、安装搭建说明修改配置文件及打开所有端口，并启动6个redis               

          2、创建redis集群是使用ruby命令，故需要安装以下软件

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | 安装 ruby 和 rubygems（注意： 需要 ruby 的版本在 1.8.7 以上）  # yum install ruby rubygems    检查 ruby 版本：  # ruby -v  ruby 1.8.7 (2013-06-27 patchlevel 374) [x86\_64-linux]    gem 安装 redis ruby 接口：  # gem install redis |

         3、执行创建集群的命令（可在任何一个redis的src目录下执行）

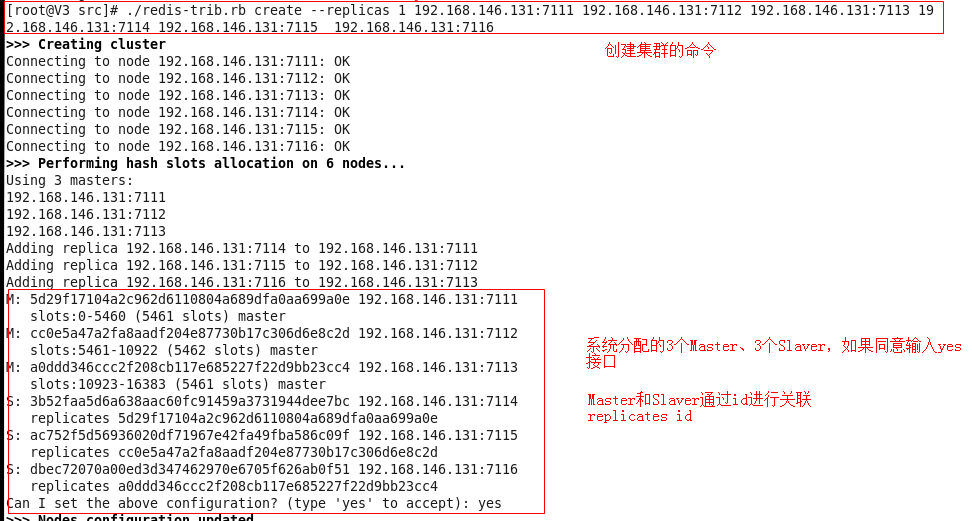
                 a) redis-trib.rb  创建集群的rube命令  
                 b) create 创建集群

                 c)  replicas 每个master包含一个slaver

                 d) 192.168.146.131:7111 每个redis实例的ip+端口，每个实例根据空格分开

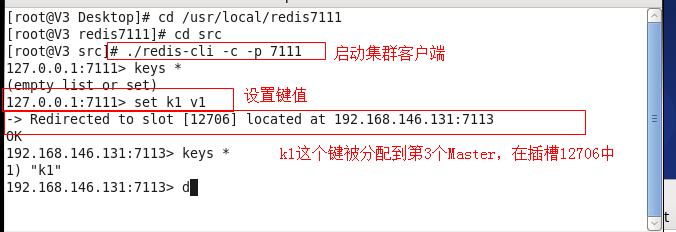
|  |  |
| --- | --- |
| 1  2  3 | cd /usr/local/redis7111/src    ./redis-trib.rb create replicas 1 192.168.146.131:7111 192.168.146.131:7112 192.168.146.131:7113 192.168.146.131:7114 192.168.146.131:7115 192.168.146.131:7116 |

            4、系统分配Master、Slaver集群关系

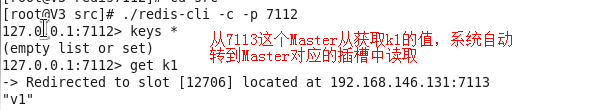


                   5、Slot插槽分配结果

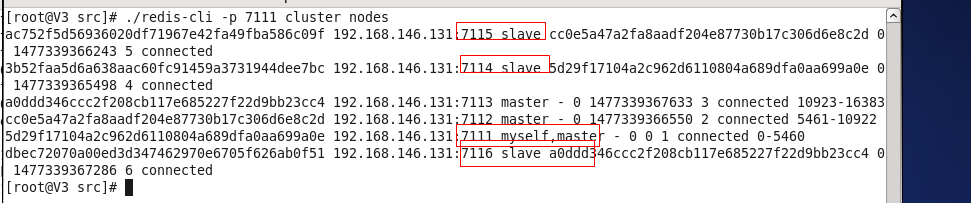
**四、集群的测试**

              1、设置key

                2、 在另外一个集群机器上读取刚设置的key



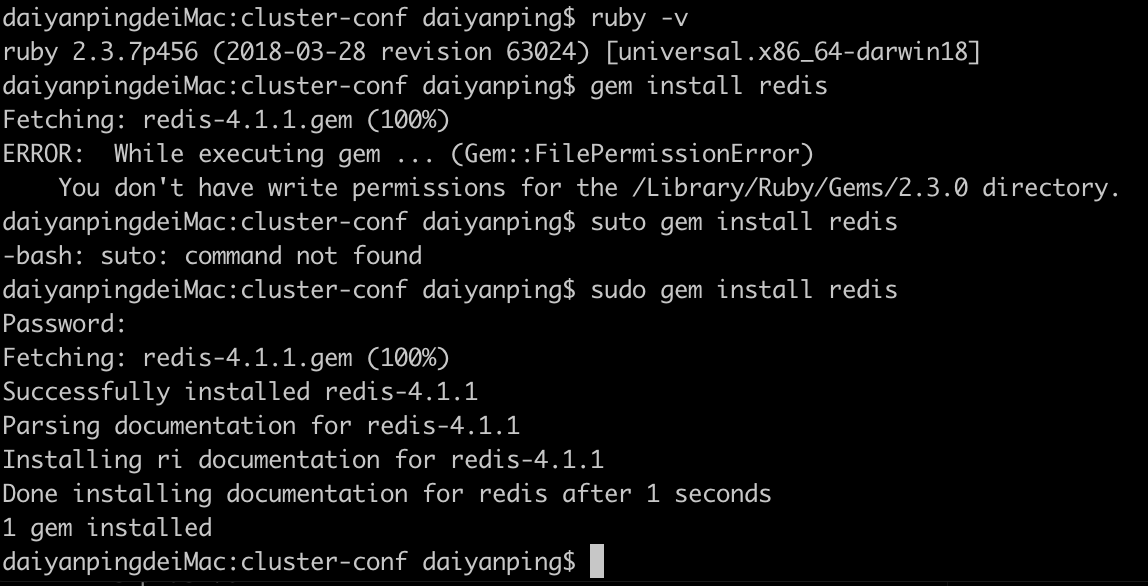
                  3、查看集群间的关系、Master/Slaver之间的关系



**五、总结**

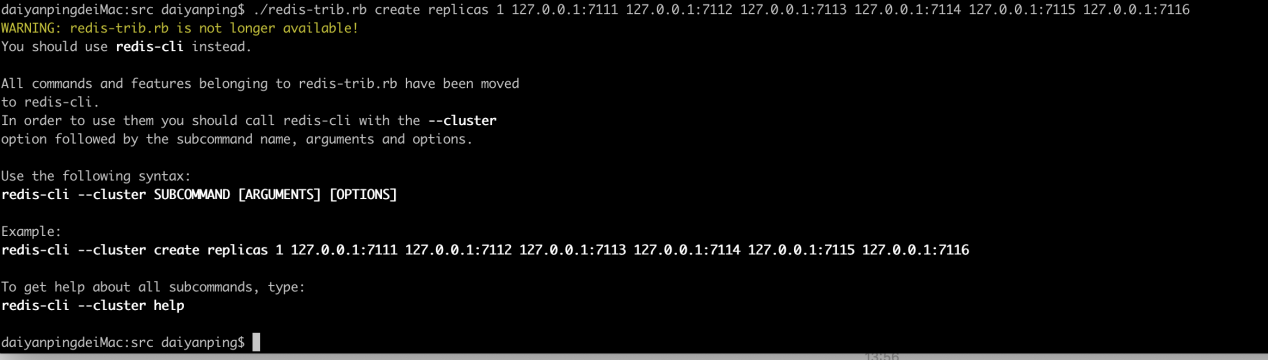
1. 集群中各台机器的配置信息一致，Master/Slaver关系是在创建集群时由系统分配
2. redis集群公用16384个slot，分配给不同的Master
3. 每个key最终都会位于某一个slot，读取key时会先转向到某一个slot，然后读取其中的值
4. 新增集群或减少集群（如新增或减少Master）会重新分配slot
5. 集群的Master/Slaver不知道怎么做容灾，自动切换等，等待进一步研究

mac 自带ruby 不用安装

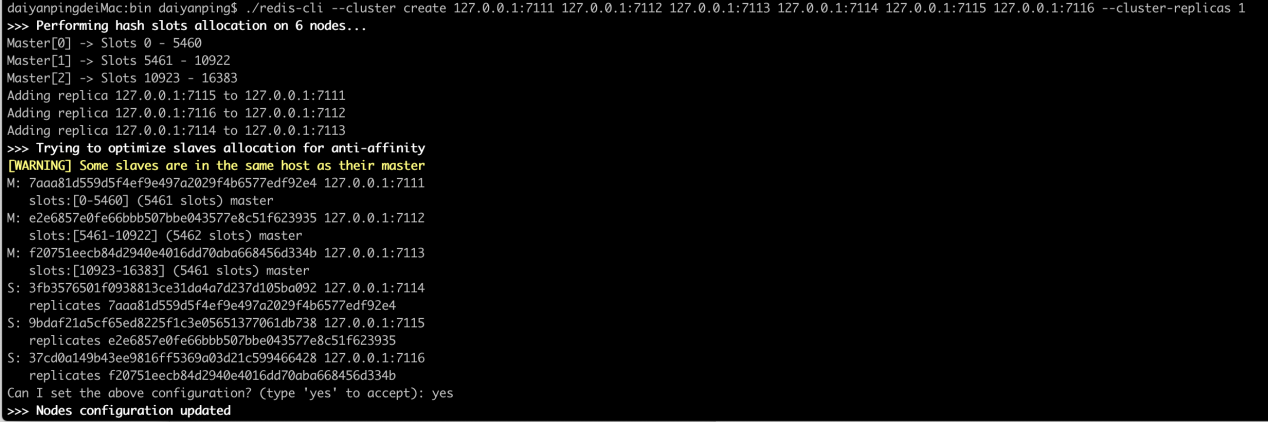


直接执行sudo gem install redis 即可

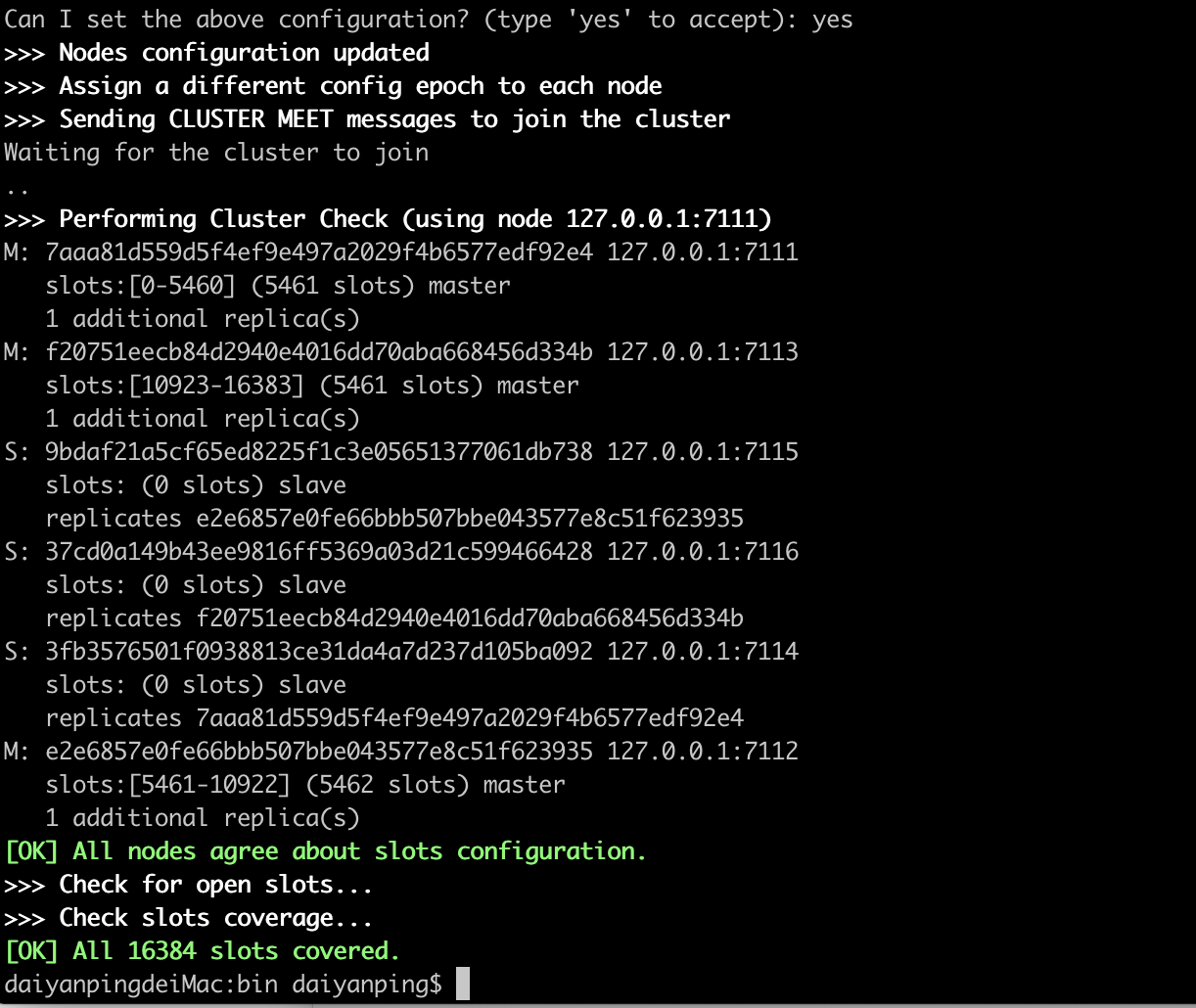
通过redis-trib.rb方式创建集群已被禁止，如下：



新的创建方式是使用Redis-cli进行命令如下：



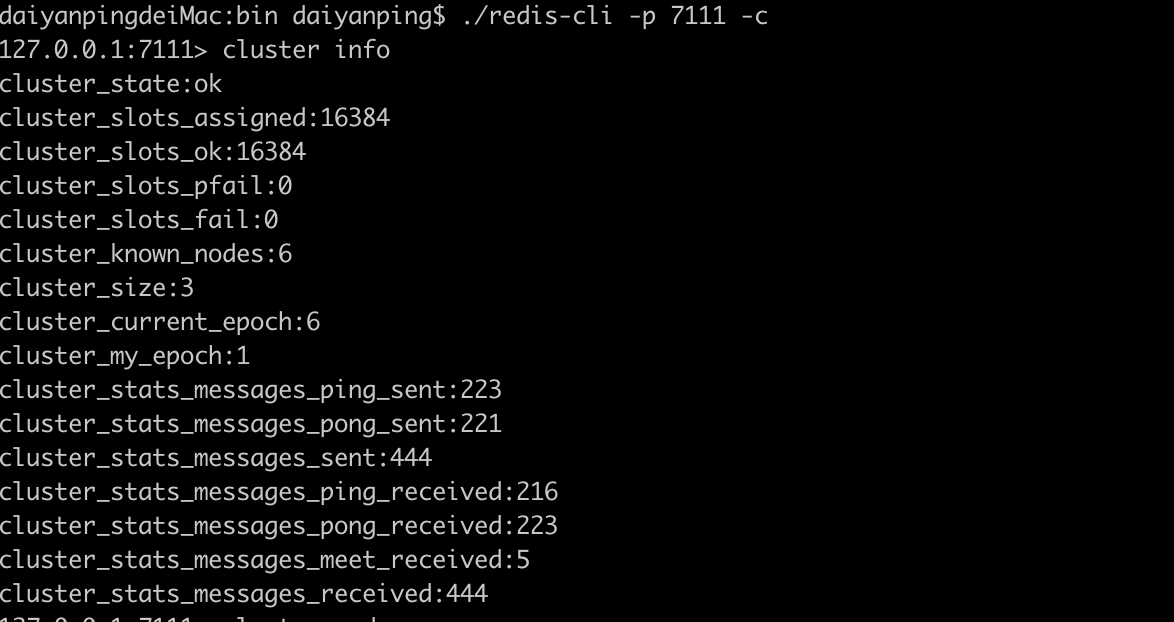
然后输入yes



可以看到3个master,3个slave

下面使用集群方式登录任意节点：

在普通的形式上加上参数-c



使用cluster info 查看集群信息

cluster\_state:ok                 #集群状态

cluster\_slots\_assigned:16384     #被分配的槽位数

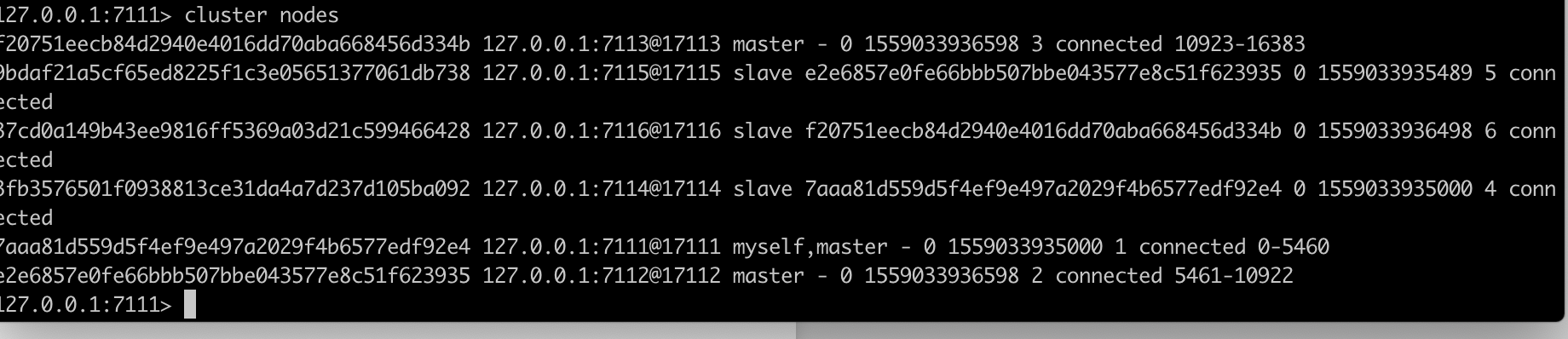
cluster\_slots\_ok:16384           #正确分配的槽位

cluster\_slots\_pfail:0

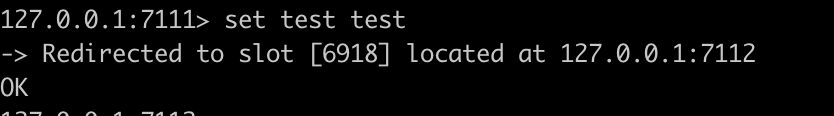
cluster\_slots\_fail:0

cluster\_known\_nodes:6            #集群节点数

使用cluster nodes 查看集群节点

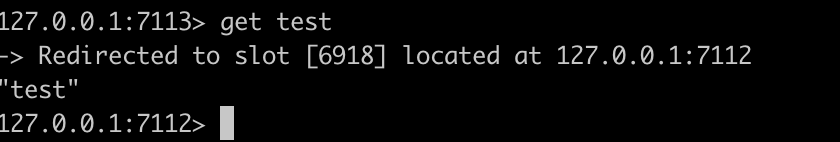


往集群中插入数据

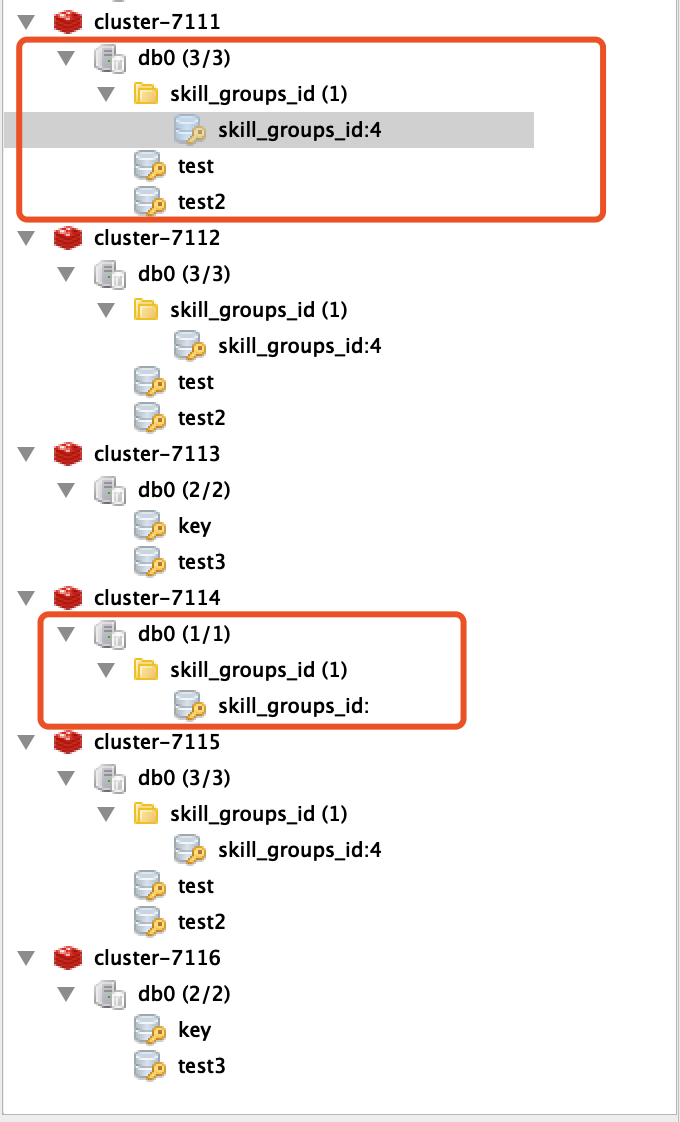


我们在连接7111的客户端上插入数据，实际上数据插入到了7112上

集群获取数据



任意一个节点都可以获取数据

测试发现，有时候，集群中的主从不同步问题

**三、集群客户端命令（redis-cli -c -p port）**

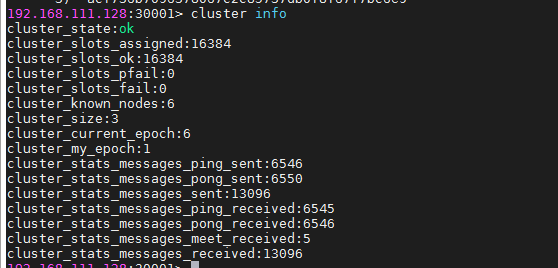
**集群**  
cluster info ：打印集群的信息  
cluster nodes ：列出集群当前已知的所有节点（ node），以及这些节点的相关信息。  
**节点**  
cluster meet <ip> <port> ：将 ip 和 port 所指定的节点添加到集群当中，让它成为集群的一份子。  
cluster forget <node\_id> ：从集群中移除 node\_id 指定的节点。  
cluster replicate <node\_id> ：将当前节点设置为 node\_id 指定的节点的从节点。  
cluster saveconfig ：将节点的配置文件保存到硬盘里面。  
**槽(slot)**cluster addslots <slot> [slot ...] ：将一个或多个槽（ slot）指派（ assign）给当前节点。  
cluster delslots <slot> [slot ...] ：移除一个或多个槽对当前节点的指派。  
cluster flushslots ：移除指派给当前节点的所有槽，让当前节点变成一个没有指派任何槽的节点。  
cluster setslot <slot> node <node\_id> ：将槽 slot 指派给 node\_id 指定的节点，如果槽已经指派给  
**另一个节点，那么先让另一个节点删除该槽>，然后再进行指派。**cluster setslot <slot> migrating <node\_id> ：将本节点的槽 slot 迁移到 node\_id 指定的节点中。  
cluster setslot <slot> importing <node\_id> ：从 node\_id 指定的节点中导入槽 slot 到本节点。  
cluster setslot <slot> stable ：取消对槽 slot 的导入（ import）或者迁移（ migrate）。  
**键**  
cluster keyslot <key> ：计算键 key 应该被放置在哪个槽上。  
cluster countkeysinslot <slot> ：返回槽 slot 目前包含的键值对数量。  
cluster getkeysinslot <slot> <count> ：返回 count 个 slot 槽中的键

注意点：集群关闭后，要想重新启动集群，必须将集群中每个节点的dump.rdb，nodes.conf文件删除掉

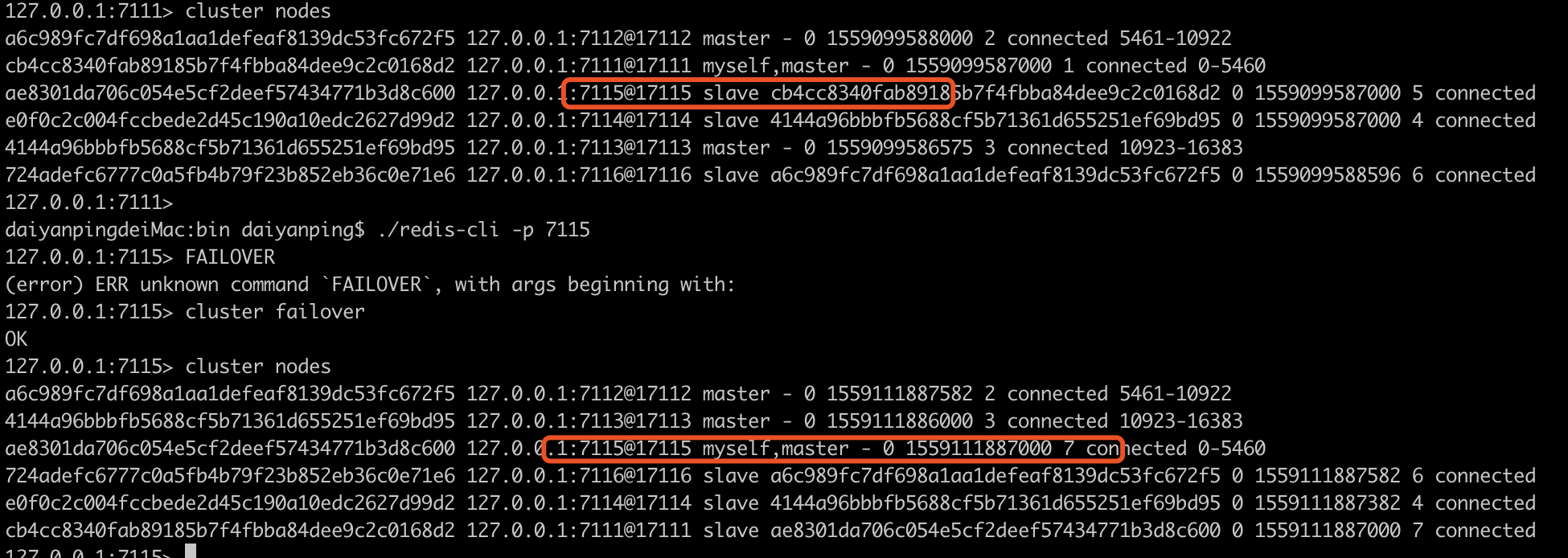
本地集群快捷搭建方式，直接进入到redis安装目录，然后找到utils目录，然后进入create-cluster目录，然后执行create-cluster脚本，即可

集群命令：

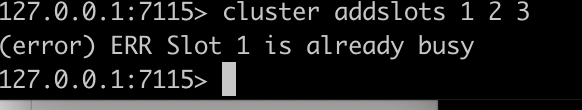
cluster info ：打印集群的信息



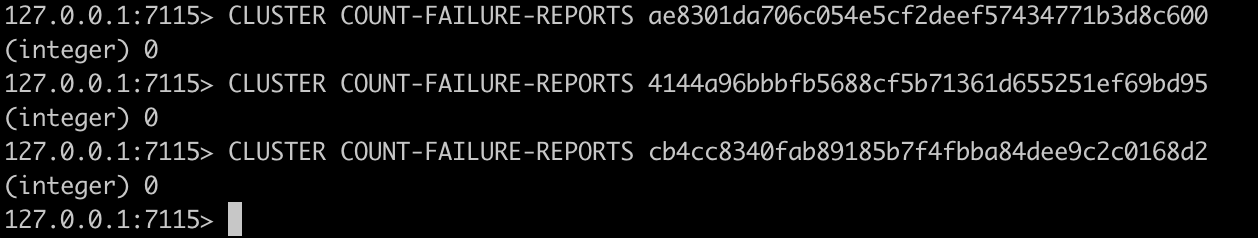
**[CLUSTER FAILOVER](http://www.redis.cn/commands/cluster-failover.html)**： 在某个从机上执行该命令，将从机与主机进行交换



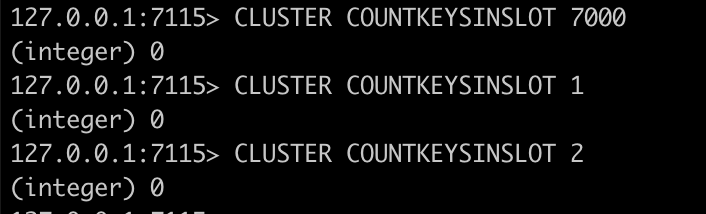
**[CLUSTER ADDSLOTS](http://www.redis.cn/commands/cluster-addslots.html)**：为某个节点分配hash槽，如果指定的hash槽已经被分配则失败。该命令一般是在初始化集群时使用，或修复损坏的hash槽时使用



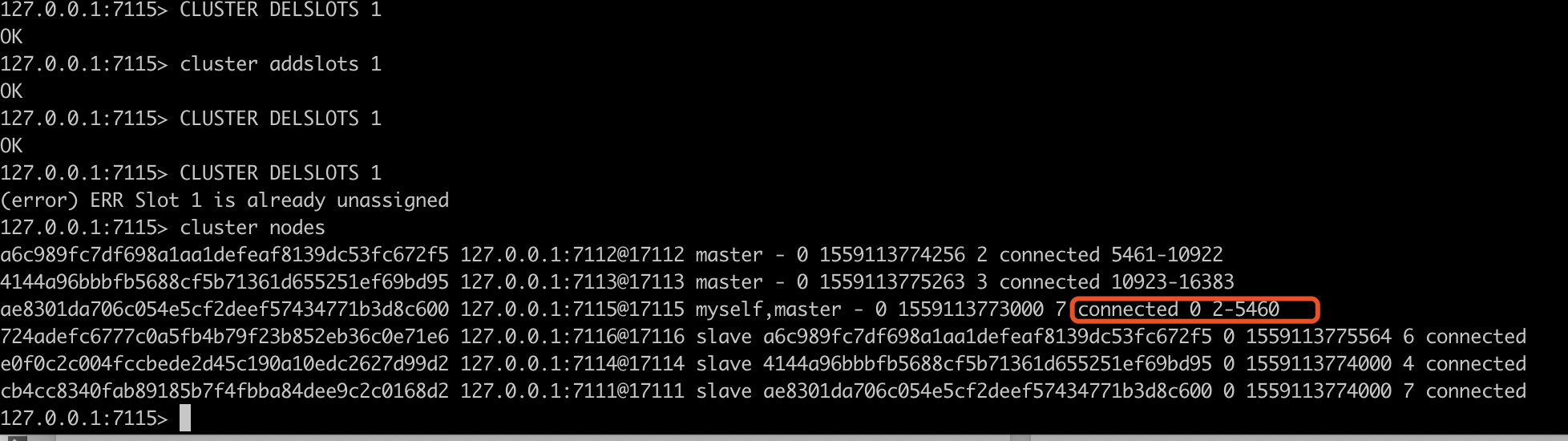
**[CLUSTER COUNT-FAILURE-REPORTS](http://www.redis.cn/commands/cluster-count-failure-reports.html)**：返回指定节点的故障报告个数，故障报告有过期时间，是节点超时时间的两倍，该命令一般用于调试



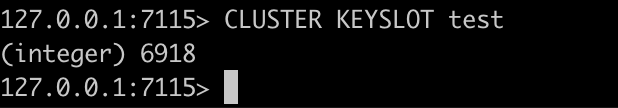
**[CLUSTER COUNTKEYSINSLOT](http://www.redis.cn/commands/cluster-countkeysinslot.html)**：返回连接节点指定hash槽上key的数量



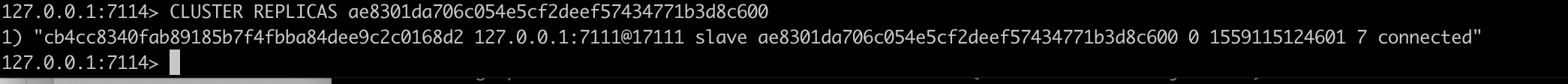
**[CLUSTER DELSLOTS](http://www.redis.cn/commands/cluster-delslots.html)**:移除某个master节点已经关联的指定hash槽

**[CLUSTER FORGET](http://www.redis.cn/commands/cluster-forget.html)**：删除连接节点上*nodes table*中指定节点,已删除的节点的节点ID被加入禁止列表，保留1分钟

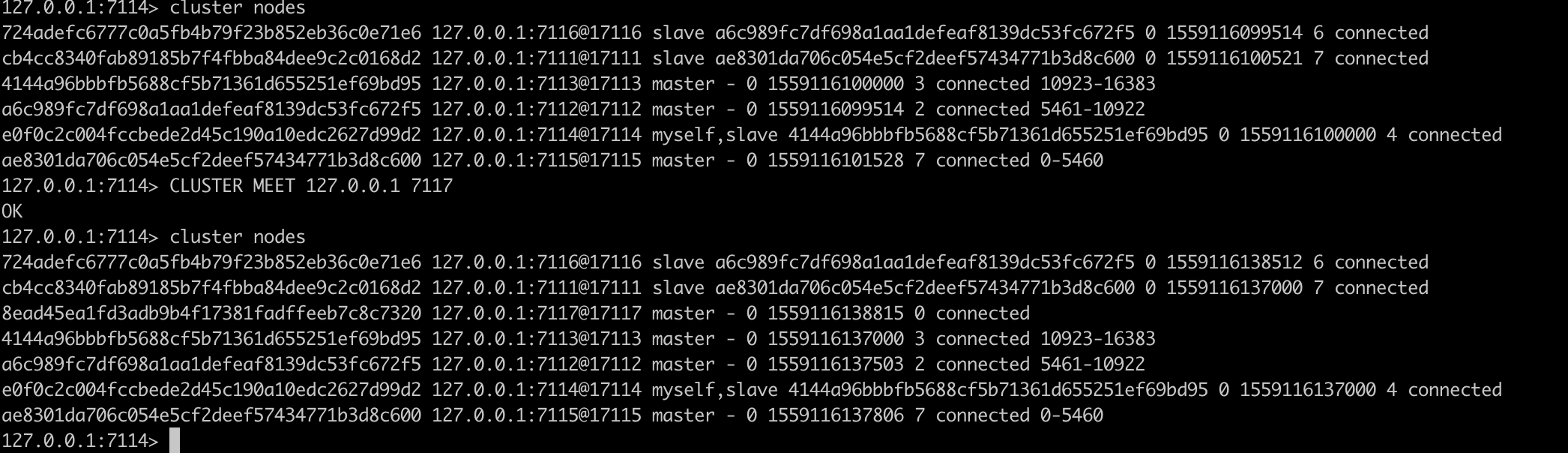
**[CLUSTER KEYSLOT](http://www.redis.cn/commands/cluster-keyslot.html)**:查询指定key，存在那个has槽



**[CLUSTER REPLICAS](http://www.redis.cn/commands/cluster-replicas.html)**:返回指定master节点的slave节点信息

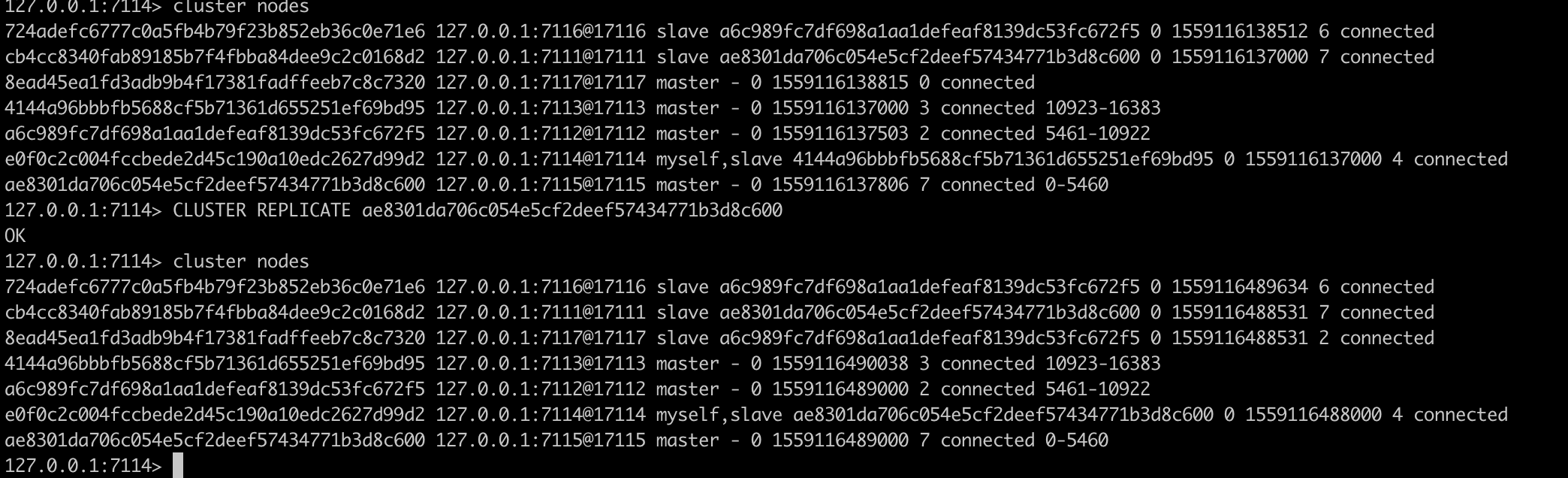


**[CLUSTER MEET](http://www.redis.cn/commands/cluster-meet.html)**：将某个节点加到集群中



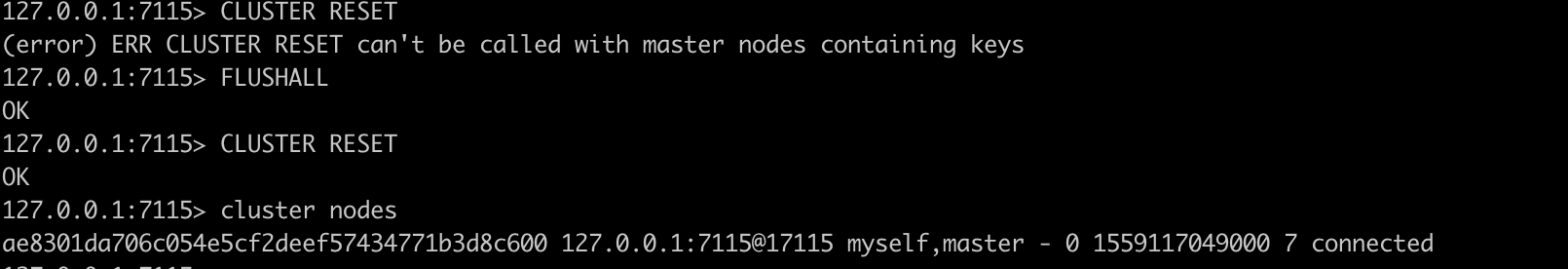
**[CLUSTER REPLICATE](http://www.redis.cn/commands/cluster-replicate.html)**：指定连接节点成为某个master节点的slave节点

一般就是修改某个slave节点的master节点，或者将没有分配hash槽的master节点变为某个master节点的slave节点



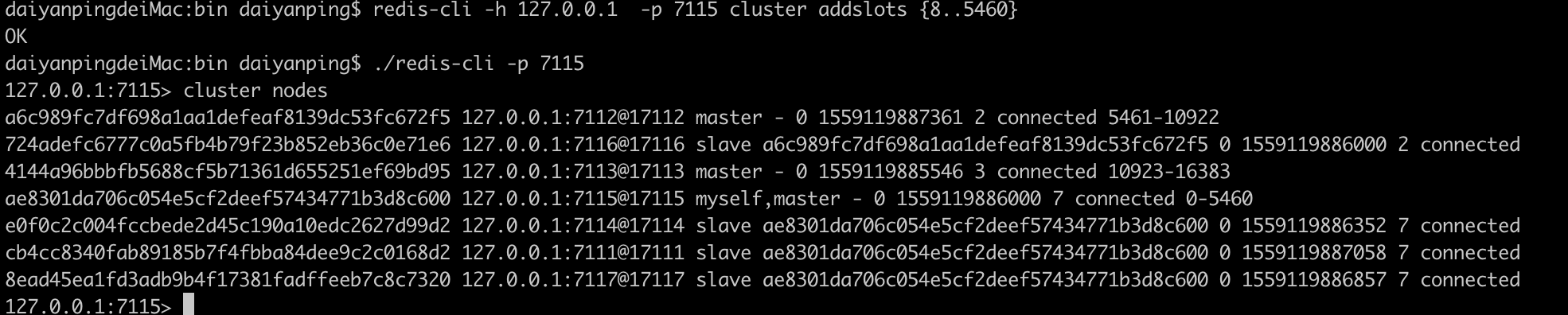
**[CLUSTER RESET](http://www.redis.cn/commands/cluster-reset.html)**：重置某个master节点，如果存在key，则需要先移除key

重置后，集群中的其他节点都会忽略，nodes命令只能看的到自己，该master将不会分配hash槽

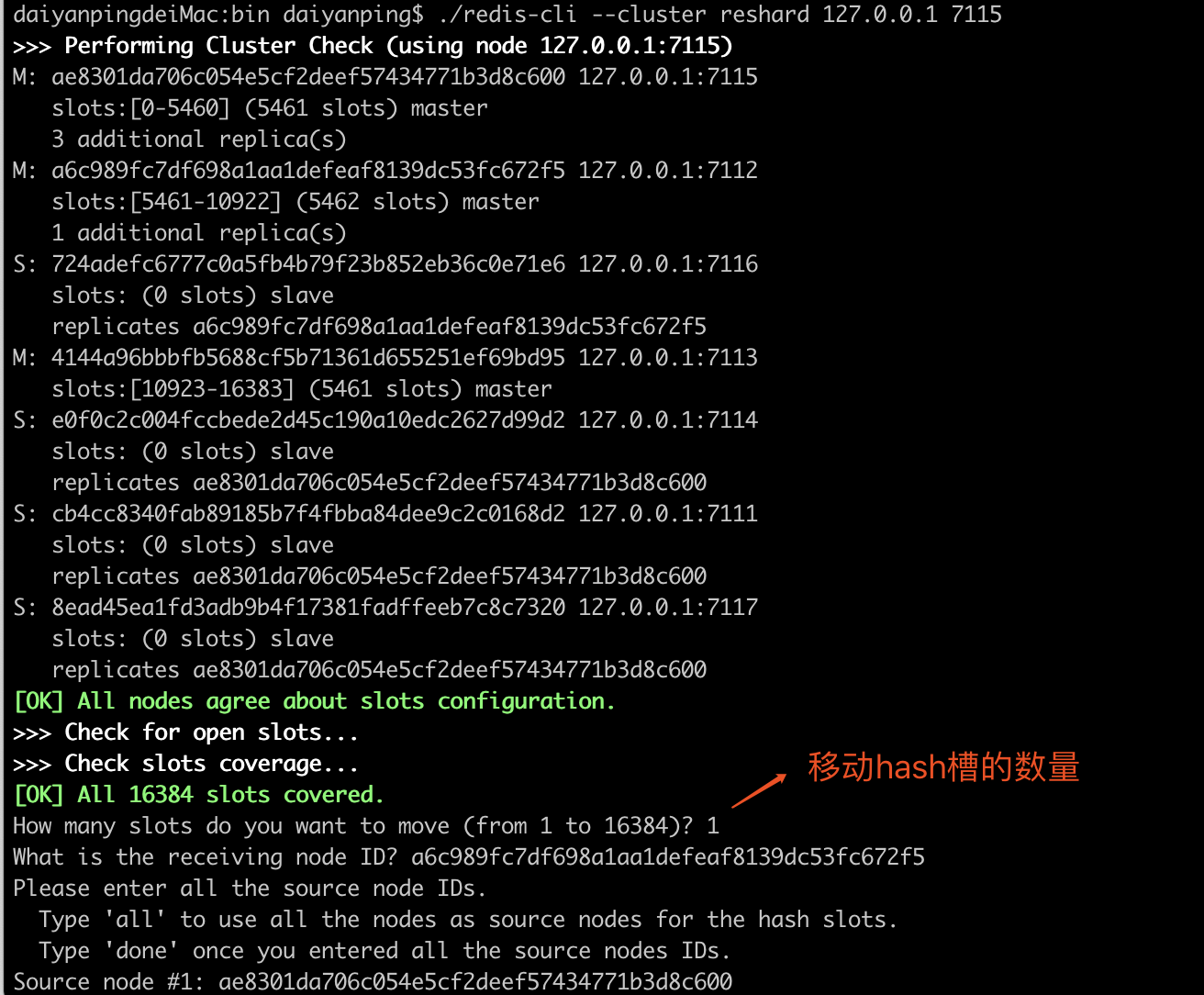


重置，可以使用meet 命令重新连接到集群，hash槽需要手动分配

手动分配hash槽：



在hash槽已经全部分配的情况下，重新分配hash槽：





这种方式只能按顺序方式移动hash槽

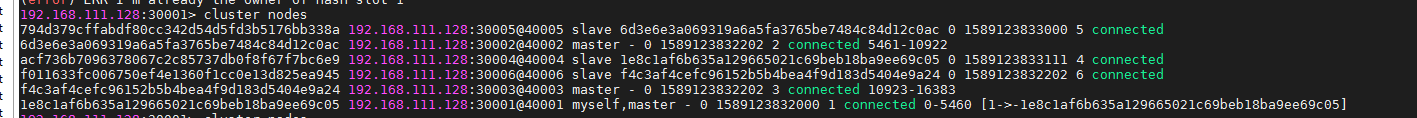
手动迁移指定的hash槽：

第一步：设置迁移节点hash槽的状态为MIGRATING



CLUSTER SETSLOT 1 MIGRATING 1e8c1af6b635a129665021c69beb18ba9ee69c

查看节点状态：



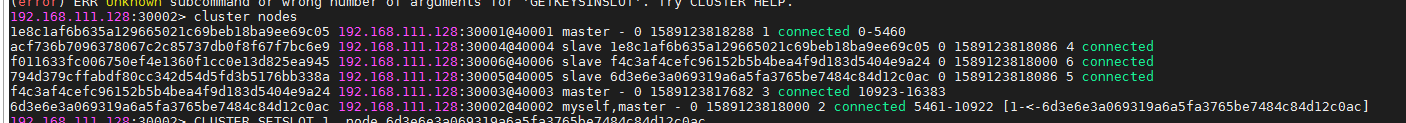
有一个hash槽迁出

第二步：设置迁入节点hash槽的状态为IMPORTING

CLUSTER SETSLOT 1 IMPORTING 6d3e6e3a069319a6a5fa3765be7484c84d12c0ac

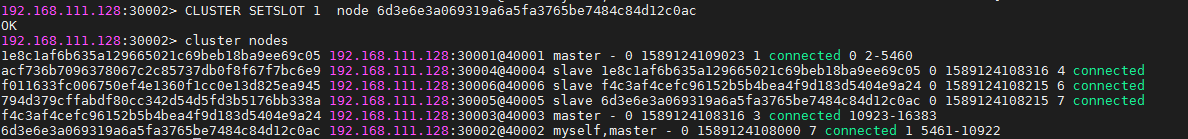


查看节点状态：



有一个hash槽迁入

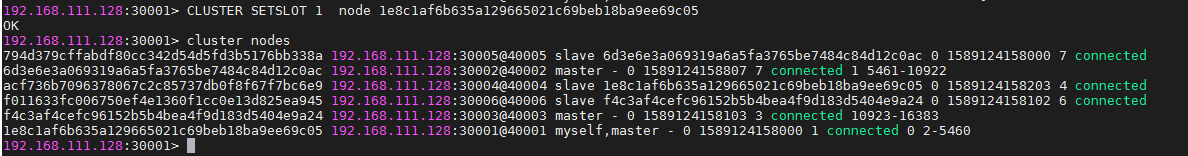
第三步：在迁入节点执行命令进hash槽迁入



查看节点状态：hash槽已成功迁入

CLUSTER SETSLOT 1 node 6d3e6e3a069319a6a5fa3765be7484c84d12c0ac

第4步：在迁出节点执行命令进行hash槽状态清除



查看节点状态：hash 槽状态已成功清除