高可用架构篇

Redis 集群的扩展测试

注意：本节教程内容紧接上两节教程《高可用架构篇--第 05 节--Redis 集群的安装（Redis3+CentOS）》和《高可用架构篇--第 06 节--Redis 集群的高可用测试（含 Jedis 客户端的使用）》的内容

一、安装新的 Redis 节点，将用于扩展性测试

1、在 192.168.1.117 虚拟机上以同样的方式安装 Redis3，并启动两个实例，规划如下：

主机名	IP	服务端口 默认 6379	集群端口 服务端口值+10000	主/从
edu-redis-07	192.168.1.117	7117	17117	Master
edu-redis-07	192.168.1.117	7118	17118	Slave

按规划：在 192.168.1.117 的防火墙中打开相应的端口

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 7117 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 17117 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 7118 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 17118 -j ACCEPT
```

2、Redis 安装过程

可请参考《高可用架构篇--第 05 节--Redis 集群的安装（Redis3+CentOS）》教程的文档，命令如下：

```
# yum install gcc tcl
# cd /usr/local/src
# wget http://download.redis.io/releases/redis-3.0.3.tar.gz
# mkdir /usr/local/redis3
# tar -zxvf redis-3.0.3.tar.gz
# cd redis-3.0.3
# make PREFIX=/usr/local/redis3 install
# yum install ruby rubygems
# gem install redis
```

3、创建集群配置目录，并拷贝 redid.conf 配置文件到各节点配置目录：

192.168.1.117

```
# mkdir -p /usr/local/redis3/cluster/7117
# mkdir -p /usr/local/redis3/cluster/7118
# cp /usr/local/src/redis-3.0.3/redis.conf /usr/local/redis3/cluster/7117/redis-7117.conf
# cp /usr/local/src/redis-3.0.3/redis.conf /usr/local/redis3/cluster/7118/redis-7118.conf
```

提示：conf 配置文件具体内容请看教程提供的 redis-7117.conf 和 redis-7118.conf 配置文件，主要增加了数据目录 dir 属性的配置。

4、在 192.168.1.117 上使用如下命令启动这 2 个 Redis 实例：

```
# /usr/local/redis3/bin/redis-server /usr/local/redis3/cluster/7117/redis-7117.conf
# /usr/local/redis3/bin/redis-server /usr/local/redis3/cluster/7118/redis-7118.conf
# ps -ef | grep redis
root      4865     1  0 01:01 ?        00:00:00 /usr/local/redis3/bin/redis-server *:7117 [cluster]
root      4869     1  0 01:01 ?        00:00:00 /usr/local/redis3/bin/redis-server *:7118 [cluster]
```

二、Redis 集群的扩展性测试

1、redis-trib.rb 命令介绍：

```
[root@edu-redis-01 src]# /usr/local/src/redis-3.0.3/src/redis-trib.rb
Usage: redis-trib <command> <options> <arguments ...>
  import          host:port
                  --from <arg>
  set-timeout     host:port milliseconds
  del-node        host:port node_id
  create          host1:port1 ... hostN:portN
                  --replicas <arg>
  help            (show this help)
  add-node        new_host:new_port existing_host:existing_port
                  --slave
                  --master-id <arg>
  reshard         host:port
                  --slots <arg>
                  --to <arg>
                  --yes
                  --from <arg>
  fix             host:port
  check           host:port
  call            host:port command arg arg .. arg
For check, fix, reshard, del-node, set-timeout you can specify the host and port of any working
node in the cluster.
```

redis-trib.rb 命令参数说明：

call：执行 redis 命令

create：创建一个新的集群（上一节教程有介绍）

add-node：将一个节点添加到集群里面，第一个是新节点 ip:port， 第二个是集群中任意一个正常节点 ip:port，--master-id

reshard：重新分片

check：查看集群信息

del-node：移除一个节点

2、添加新的 Master 节点：

add-node 将一个节点添加到集群里面，第一个是新节点 ip:port，第二个是任意一个已存在节点 ip:port

```
# /usr/local/src/redis-3.0.3/src/redis-trib.rb add-node 192.168.1.117:7117 192.168.1.111:7111
>>> Adding node 192.168.1.117:7117 to cluster 192.168.1.111:7111
Connecting to node 192.168.1.111:7111: OK
Connecting to node 192.168.1.116:7116: OK
Connecting to node 192.168.1.113:7113: OK
Connecting to node 192.168.1.112:7112: OK
Connecting to node 192.168.1.115:7115: OK
Connecting to node 192.168.1.114:7114: OK
```

```
>>> Performing Cluster Check (using node 192.168.1.111:7111)
M: cc50047487b52697d62b1a72b231b7c74e08e051 192.168.1.111:7111
   slots:10923-16383 (5461 slots) master
   1 additional replica(s)
S: b21ae6d0a3e614e53bbc52639173ec3ad68044b5 192.168.1.116:7116
   slots: (0 slots) slave
   replicates 041addd95fa0a15d98be363034e53dd06f69ef47
M: 041addd95fa0a15d98be363034e53dd06f69ef47 192.168.1.113:7113
   slots:0-5460 (5461 slots) master
   1 additional replica(s)
M: 712e523b617eea5a2ed8df732a50ff298ae2ea48 192.168.1.112:7112
   slots:5461-10922 (5462 slots) master
   1 additional replica(s)
S: 55c0db5af1b917f3ce0783131fb8bab28920e1f3 192.168.1.115:7115
   slots: (0 slots) slave
   replicates 712e523b617eea5a2ed8df732a50ff298ae2ea48
S: 8a6ca1452d61f8b4726f0649e6ce49a6ec4afee2 192.168.1.114:7114
   slots: (0 slots) slave
   replicates cc50047487b52697d62b1a72b231b7c74e08e051
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
Connecting to node 192.168.1.117:7117: OK
>>> Send CLUSTER MEET to node 192.168.1.117:7117 to make it join the cluster.
[OK] New node added correctly.
```

以上操作结果表示节点添加成功，新增的节点不包含任何数据， 因为它没有分配任何 slot。
新加入的节点是一个 master 节点，当集群需要将某个从节点升级为新的主节点时，这个新节点不会被选中。

为新节点分配哈希槽（slot）：
你只需要指定集群中其中一个节点的地址，redis-trib 就会自动找到集群中的其他节点。目前 redis-trib 只能在管理员的协助下完成重新分片的工作，命令如下：

```
# /usr/local/src/redis-3.0.3/src/redis-trib.rb reshard 192.168.1.111:7111
Connecting to node 192.168.1.111:7111: OK
Connecting to node 192.168.1.117:7117: OK
Connecting to node 192.168.1.116:7116: OK
Connecting to node 192.168.1.113:7113: OK
Connecting to node 192.168.1.112:7112: OK
Connecting to node 192.168.1.115:7115: OK
Connecting to node 192.168.1.114:7114: OK
>>> Performing Cluster Check (using node 192.168.1.111:7111)
M: cc50047487b52697d62b1a72b231b7c74e08e051 192.168.1.111:7111
```

```
   slots:10923-16383 (5461 slots) master
   1 additional replica(s)
M: badbc0ffde2a3700df7e179d23fa2762108eabba 192.168.1.117:7117
   slots: (0 slots) master
   0 additional replica(s)
S: b21ae6d0a3e614e53bbc52639173ec3ad68044b5 192.168.1.116:7116
   slots: (0 slots) slave
   replicates 041addd95fa0a15d98be363034e53dd06f69ef47
M: 041addd95fa0a15d98be363034e53dd06f69ef47 192.168.1.113:7113
   slots:0-5460 (5461 slots) master
   1 additional replica(s)
M: 712e523b617eea5a2ed8df732a50ff298ae2ea48 192.168.1.112:7112
   slots:5461-10922 (5462 slots) master
   1 additional replica(s)
S: 55c0db5af1b917f3ce0783131fb8bab28920e1f3 192.168.1.115:7115
   slots: (0 slots) slave
   replicates 712e523b617eea5a2ed8df732a50ff298ae2ea48
S: 8a6ca1452d61f8b4726f0649e6ce49a6ec4afee2 192.168.1.114:7114
   slots: (0 slots) slave
   replicates cc50047487b52697d62b1a72b231b7c74e08e051
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
How many slots do you want to move (from 1 to 16384)? 500
```

上面的提示是确认你打算移动的哈希槽 slots 的数量（这里槽数量设置为 500）

除了移动的哈希槽数量之外， redis-trib 还需要知道重新分片的目标（target node)， 也就是负责接收这 500 个哈希槽的节点。指定目标需要使用节点的 ID ， 而不是 IP 地址和端口。我们打算向上面新增的主节点来作为目标， 它的 IP 地址和端口是 192.168.1.117:7117，而节点 ID 则是 badbc0ffde2a3700df7e179d23fa2762108eabba ， 那么我们应该向 redis-trib 提供节点的 ID ：

```
What is the receiving node ID? badbc0ffde2a3700df7e179d23fa2762108eabba
```

接下来 redis-trib 会向你询问重新分片的源节点（source node)， 也就是要从哪个节点中取出 500 个哈希槽，并将这些槽移动到目标节点上面。如果我们不打算从特定的节点上取出指定数量的哈希槽， 那么可以向 redis-trib 输入 all ， 这样的话， 集群中的所有主节点都会成为源节点， redis-trib 将从各个源节点中各取出一部分哈希槽， 凑够 500 个， 然后移动到目标节点上面：

```
Please enter all the source node IDs.
  Type 'all' to use all the nodes as source nodes for the hash slots.
  Type 'done' once you entered all the source nodes IDs.
Source node #1:all
```

输入 all 并按下回车之后， redis-trib 将打印出哈希槽的移动计划：

```
Do you want to proceed with the proposed reshard plan (yes/no)? yes
```

如果你觉得没问题的话， 就可以输入 yes 并再次按回车确认， redis-trib 就会正式开始执行重新分片操作， 将指定的哈希槽从源节点一个个地移动到目标节点上面。

注意：可以同步观察重新分片是否会对客户端的连续使用产生影响（结果：不影响）。

移动前，7117 上没有 slot：

```
127.0.0.1:7116> cluster nodes
f34b28f1483f0c0d9543e93938fc12b8818050cb 192.168.1.115:7115 slave d2c6c159b07e8197e2c8d2eae8c847050159f602 0 1439137429295 5 connected
ab31611b3424990e2b9bbe73135cb4cb0ace394f 192.168.1.117:7117 master - 0 1439137431299 0 connected
1fd90d54090925afb4087d4ef94a1710a25160d6 192.168.1.116:7116 myself,slave 4e46bd06654e8660e617f7249fa22f6fa1fdff0d 0 0 3 connected
48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114 master - 0 1439137428292 9 connected 10923-16383
8dd55e9b4da9f62b9b15232e86553f1337864179 192.168.1.111:7111 slave 48db78bcc55c4c3a3788940a6458b921ccf95d44 0 1439137431798 9 connected
d2c6c159b07e8197e2c8d2eae8c847050159f602 192.168.1.112:7112 master - 0 1439137430296 5 connected 5461-10922
4e46bd06654e8660e617f7249fa22f6fa1fdff0d 192.168.1.113:7113 master - 0 1439137427291 6 connected 0-5460
```

移动后，7117 上有 3 段 slot：

```
127.0.0.1:7116> cluster nodes
f34b28f1483f0c0d9543e93938fc12b8818050cb 192.168.1.115:7115 slave d2c6c159b07e8197e2c8d2eae8c847050159f602 0 1439138042569 5 connected
ab31611b3424990e2b9bbe73135cb4cb0ace394f 192.168.1.117:7117 master - 0 1439138041567 10 connected 0-165 5461-5627 10923-11088
1fd90d54090925afb4087d4ef94a1710a25160d6 192.168.1.116:7116 myself,slave 4e46bd06654e8660e617f7249fa22f6fa1fdff0d 0 0 3 connected
48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114 master - 0 1439138043572 9 connected 11089-16383
8dd55e9b4da9f62b9b15232e86553f1337864179 192.168.1.111:7111 slave 48db78bcc55c4c3a3788940a6458b921ccf95d44 0 1439138041066 9 connected
d2c6c159b07e8197e2c8d2eae8c847050159f602 192.168.1.112:7112 master - 0 1439138044574 5 connected 5628-10922
4e46bd06654e8660e617f7249fa22f6fa1fdff0d 192.168.1.113:7113 master - 0 1439138045576 6 connected 166-5460
127.0.0.1:7116>
```

在重新分片操作执行完毕之后， 可以使用以下命令来检查集群是否正常：

```
# /usr/local/src/redis-3.0.3/src/redis-trib.rb check 192.168.1.111:7111
```

```
[root@edu-redis-01 7111]# /usr/local/src/redis-3.0.3/src/redis-trib.rb check 192.168.1.111:7111
Connecting to node 192.168.1.111:7111: OK
Connecting to node 192.168.1.117:7117: OK
Connecting to node 192.168.1.116:7116: OK
Connecting to node 192.168.1.113:7113: OK
Connecting to node 192.168.1.114:7114: OK
Connecting to node 192.168.1.112:7112: OK
Connecting to node 192.168.1.115:7115: OK
>>> Performing Cluster Check (using node 192.168.1.111:7111)
S: 8dd55e9b4da9f62b9b15232e86553f1337864179 192.168.1.111:7111
   slots: (0 slots) slave
   replicates 48db78bcc55c4c3a3788940a6458b921ccf95d44
M: ab31611b3424990e2b9bbe73135cb4cb0ace394f 192.168.1.117:7117
   slots:0-165,5461-5627,10923-11088 (499 slots) master
   0 additional replica(s)
S: 1fd90d54090925afb4087d4ef94a1710a25160d6 192.168.1.116:7116
   slots: (0 slots) slave
   replicates 4e46bd06654e8660e617f7249fa22f6fa1fdff0d
M: 4e46bd06654e8660e617f7249fa22f6fa1fdff0d 192.168.1.113:7113
   slots:166-5460 (5295 slots) master
   1 additional replica(s)
M: 48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114
   slots:11089-16383 (5295 slots) master
   1 additional replica(s)
M: d2c6c159b07e8197e2c8d2eae8c847050159f602 192.168.1.112:7112
   slots:5628-10922 (5295 slots) master
   1 additional replica(s)
S: f34b28f1483f0c0d9543e93938fc12b8818050cb 192.168.1.115:7115
   slots: (0 slots) slave
   replicates d2c6c159b07e8197e2c8d2eae8c847050159f602
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
```

上面输出的检查结果显示，重新分片成功，集群状态正常。

也可以用以下命令再次查看集群的节点状况：

```
# /usr/local/redis3/bin/redis-cli -c -p 7111 cluster nodes
```

```
[root@edu-redis-01 ~]# /usr/local/redis3/bin/redis-cli -c -p 7111 cluster nodes
badbc0ffde2a3700df7e179d23fa2762108eabba 192.168.1.117:7117 master - 0 1438880689365 7 connected 0-165 5461-5627 10923-11088
b21ae6d0a3e614e53bbc52639173ec3ad68044b5 192.168.1.116:7116 slave 041addd95fa0a15d98be363034e53dd06f69ef47 0 1438880694376 6 connected
041addd95fa0a15d98be363034e53dd06f69ef47 192.168.1.113:7113 master - 0 1438880695379 6 connected 166-5460
712e523b617eea5a2ed8df732a50ff298ae2ea48 192.168.1.112:7112 master - 0 1438880693374 5 connected 5628-10922
55c0db5af1b917f3ce0783131fb8bab28920e1f3 192.168.1.115:7115 slave 712e523b617eea5a2ed8df732a50ff298ae2ea48 0 1438880691370 5 connected
8a6ca1452d61f8b4726f0649e6ce49a6ec4afee2 192.168.1.114:7114 slave cc50047487b52697d62b1a72b231b7c74e08e051 0 1438880692371 4 connected
cc50047487b52697d62b1a72b231b7c74e08e051 192.168.1.111:7111 myself,master - 0 0 4 connected 11089-16383
```

以上集群状态输出信息解析:

(1)节点 ID

(2)IP:PORT

(3)节点状态标识: master、slave、myself、fail?、fail

(4)如果是从节点，表示主节点的 ID；如果是主节点，则为 '-'

(5)集群最近一次向各个节点发送 PING 命令后，过去多长时间还没有接到回复

(6)节点最近一次返回 PONG 的时间戳

(7)节点的配置纪元

(8)本节点的网络连接情况: connected、disconnected

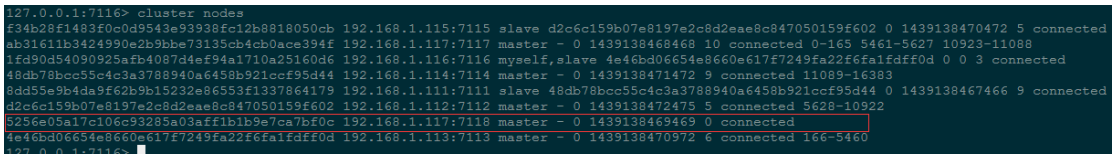
(9)如果是主节点，表示节点包含的槽

3、添加新的 slave 节点

(1)添加节点

```
#/usr/local/src/redis-3.0.3/src/redis-trib.rb add-node 192.168.1.117:7118 192.168.1.111:7111
>>> Adding node 192.168.1.117:7118 to cluster 192.168.1.111:7111
Connecting to node 192.168.1.111:7111: OK
Connecting to node 192.168.1.117:7117: OK
Connecting to node 192.168.1.116:7116: OK
Connecting to node 192.168.1.113:7113: OK
Connecting to node 192.168.1.112:7112: OK
Connecting to node 192.168.1.115:7115: OK
Connecting to node 192.168.1.114:7114: OK
>>> Performing Cluster Check (using node 192.168.1.111:7111)
M: cc50047487b52697d62b1a72b231b7c74e08e051 192.168.1.111:7111
   slots:11089-16383 (5295 slots) master
   1 additional replica(s)
M: badbc0ffde2a3700df7e179d23fa2762108eabba 192.168.1.117:7117
   slots:0-165,5461-5627,10923-11088 (499 slots) master
   0 additional replica(s)
S: b21ae6d0a3e614e53bbc52639173ec3ad68044b5 192.168.1.116:7116
   slots: (0 slots) slave
   replicates 041addd95fa0a15d98be363034e53dd06f69ef47
M: 041addd95fa0a15d98be363034e53dd06f69ef47 192.168.1.113:7113
   slots:166-5460 (5295 slots) master
   1 additional replica(s)
M: 712e523b617eea5a2ed8df732a50ff298ae2ea48 192.168.1.112:7112
   slots:5628-10922 (5295 slots) master
   1 additional replica(s)
S: 55c0db5af1b917f3ce0783131fb8bab28920e1f3 192.168.1.115:7115
   slots: (0 slots) slave
   replicates 712e523b617eea5a2ed8df732a50ff298ae2ea48
S: 8a6ca1452d61f8b4726f0649e6ce49a6ec4afee2 192.168.1.114:7114
   slots: (0 slots) slave
   replicates cc50047487b52697d62b1a72b231b7c74e08e051
```

```
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
Connecting to node 192.168.1.117:7118: OK
>>> Send CLUSTER MEET to node 192.168.1.117:7118 to make it join the cluster.
[OK] New node added correctly.
```



新增的 7118 为一个 master

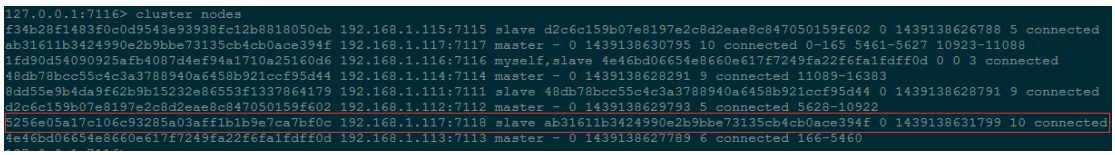
(2)redis-cli 连接上新节点 shell,输入命令:cluster replicate 对应 master 的 node-id

```
# /usr/local/redis3/bin/redis-cli -c -p 7118
127.0.0.1:7118>cluster replicate ab31611b3424990e2b9bbe73135cb4cb0ace394f
OK
```

在线添加 slave 时，需要 dump 整个 master 进程，并传递到 slave，再由 slave 加载 rdb 文件到内存，rdb 传输过程中 Master 可能无法提供服务,整个过程消耗大量 IO，因此要小心操作。

查看执行结果：

```
127.0.0.1:7116> cluster nodes
```



这时 7117 已变成 7118 的 slave

4、在线 reshard 数据

对于负载/数据不均匀的情况，可以在线 reshard slot 来解决,方法与添加新 master 的 reshard 一样，只是需要 reshard 的 master 节点是老节点。

5、删除一个 slave 节点

```
# cd /usr/local/src/redis-3.0.3/src/
# ./redis-trib.rb del-node 192.168.1.117:7118 5256e05a17c106c93285a03aff1b1b9e7ca7bf0c
```

这时候，用下以下再查看集群状态，会发现该 slave 节点已成功移除

```
# /usr/local/redis3/bin/redis-cli -c -p 7116 cluster nodes
```

```
127.0.0.1:7116> cluster nodes
f34b28f1483f0c0d9543e93938fc12b8818050cb 192.168.1.115:7115 slave d2c6c159b07e8197e2c8d2eae8c847050159f602 0 1439139021562 5 connected
ab31611b3424990e2b9bbe73135cb4cb0ace394f 192.168.1.117:7117 master - 0 1439139025568 10 connected 0-165 5461-5627 10923-11088
1fd90d54090925afb4087d4ef94a1710a25160d6 192.168.1.116:7116 myself,slave 4e46bd06654e8660e617f7249fa22f6fa1fdff0d 0 0 3 connected
48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114 master - 0 1439139023564 9 connected 11089-16383
8dd55e9b4da9f62b9b15232e86553f1337864179 192.168.1.111:7111 slave 48db78bcc55c4c3a3788940a6458b921ccf95d44 0 1439139027572 9 connected
d2c6c159b07e8197e2c8d2eae8c847050159f602 192.168.1.112:7112 master - 0 1439139024566 5 connected 5628-10922
4e46bd06654e8660e617f7249fa22f6fa1fdff0d 192.168.1.113:7113 master - 0 1439139026571 6 connected 166-5460
127.0.0.1:7116>
```

移除一个节点，对应的节点进程也会被关闭。

6、删除一个 master 节点

删除 master 节点之前首先要使用 reshard 移除该 master 的全部 slot,然后再删除当前节点(目前只能把被删除 master 的 slot 迁移到一个节点上)，操作和分配 slot 类似，指定具体的 Source node 即可。

```
# /usr/local/src/redis-3.0.3/src/redis-trib.rb reshard 192.168.1.117:7117
Connecting to node 192.168.1.117:7117: OK
Connecting to node 192.168.1.112:7112: OK
Connecting to node 192.168.1.115:7115: OK
Connecting to node 192.168.1.114:7114: OK
Connecting to node 192.168.1.111:7111: OK
Connecting to node 192.168.1.116:7116: OK
Connecting to node 192.168.1.113:7113: OK
>>> Performing Cluster Check (using node 192.168.1.117:7117)
M: ab31611b3424990e2b9bbe73135cb4cb0ace394f 192.168.1.117:7117
   slots:0-165,5461-5627,10923-11088 (499 slots) master
   0 additional replica(s)
M: d2c6c159b07e8197e2c8d2eae8c847050159f602 192.168.1.112:7112
   slots:5628-10922 (5295 slots) master
   1 additional replica(s)
S: f34b28f1483f0c0d9543e93938fc12b8818050cb 192.168.1.115:7115
   slots: (0 slots) slave
   replicates d2c6c159b07e8197e2c8d2eae8c847050159f602
M: 48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114
   slots:11089-16383 (5295 slots) master
   1 additional replica(s)
S: 8dd55e9b4da9f62b9b15232e86553f1337864179 192.168.1.111:7111
   slots: (0 slots) slave
   replicates 48db78bcc55c4c3a3788940a6458b921ccf95d44
S: 1fd90d54090925afb4087d4ef94a1710a25160d6 192.168.1.116:7116
   slots: (0 slots) slave
   replicates 4e46bd06654e8660e617f7249fa22f6fa1fdff0d
M: 4e46bd06654e8660e617f7249fa22f6fa1fdff0d 192.168.1.113:7113
   slots:166-5460 (5295 slots) master
   1 additional replica(s)
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
//输入被删除 master 的所有 slot 数量
```

```
How many slots do you want to move (from 1 to 16384)? 499
//接收 slot 的 master 节点 ID
What is the receiving node ID? 48db78bcc55c4c3a3788940a6458b921ccf95d44
Please enter all the source node IDs.
  Type 'all' to use all the nodes as source nodes for the hash slots.
  Type 'done' once you entered all the source nodes IDs.
//准备被删除 master 节点的 node-id
Source node #1: ab31611b3424990e2b9bbe73135cb4cb0ace394f
Source node #2:done


    Ready to move 499 slots.
      Source nodes:
        M: ab31611b3424990e2b9bbe73135cb4cb0ace394f 192.168.1.117:7117
       slots:0-165,5461-5627,10923-11088 (499 slots) master
       0 additional replica(s)
      Destination node:
        M: 48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114
       slots:11089-16383 (5295 slots) master
       1 additional replica(s)
      Resharding plan:
        Moving slot 0 from ab31611b3424990e2b9bbe73135cb4cb0ace394f
        Moving slot 1 from ab31611b3424990e2b9bbe73135cb4cb0ace394f
        ......
//输入 yes 执行 reshard
Do you want to proceed with the proposed reshard plan (yes/no)? yes
```

移除该 Master 节点的所有 slot 后，重新查看集群状态，会发现该节点不再占用 slot：

```
# /usr/local/redis3/bin/redis-cli -c -p 7116 cluster nodes
```

```
127.0.0.1:7116> cluster nodes
f34b28f1483f0c0d9543e93938fc12b8818050cb 192.168.1.115:7115 slave d2c6c159b07e8197e2c8d2eae8c847050159f602 0 1439139497511 5 connected
ab31611b3424990e2b9bbe73135cb4cb0ace394f 192.168.1.117:7117 master - 0 1439139498513 10 connected
1fd90d54090925afb4087d4ef94a1710a25160d6 192.168.1.116:7116 myself,slave 4e46bd06654e8660e617f7249fa22f6fa1fdff0d 0 0 3 connected
48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114 master - 0 1439139498012 11 connected 0-165 5461-5627 10923-16383
8dd55e9b4da9f62b9b15232e86553f1337864179 192.168.1.111:7111 slave 48db78bcc55c4c3a3788940a6458b921ccf95d44 0 1439139496510 11 connected
d2c6c159b07e8197e2c8d2eae8c847050159f602 192.168.1.112:7112 master - 0 1439139494508 5 connected 5628-10922
4e46bd06654e8660e617f7249fa22f6fa1fdff0d 192.168.1.113:7113 master - 0 1439139499515 6 connected 166-5460
127.0.0.1:7116>
```

确认已清空该 Master 节点的所有 slot 后就可以删除该节点了（命令与删除 slave 节点一样）：

```
# cd /usr/local/src/redis-3.0.3/src/
# ./redis-trib.rb del-node 192.168.1.117:7117 ab31611b3424990e2b9bbe73135cb4cb0ace394f
```

```
[root@edu-redis-01 src]# ./redis-trib.rb del-node 192.168.1.117:7117 ab31611b3424990e2b9bbe73135cb4cb0ace394f
>>> Removing node ab31611b3424990e2b9bbe73135cb4cb0ace394f from cluster 192.168.1.117:7117
Connecting to node 192.168.1.117:7117: OK
Connecting to node 192.168.1.112:7112: OK
Connecting to node 192.168.1.115:7115: OK
Connecting to node 192.168.1.114:7114: OK
Connecting to node 192.168.1.111:7111: OK
Connecting to node 192.168.1.116:7116: OK
Connecting to node 192.168.1.113:7113: OK
>>> Sending CLUSTER FORGET messages to the cluster...
>>> SHUTDOWN the node.
[root@edu-redis-01 src]#
```

此时执行 `ps -ef | grep redis` 命令会发现，被移除的节点对应用的 Redis 实例也已经被关闭了

```
127.0.0.1:7116> cluster nodes
f34b28f1483f0c0d9543e93938fc12b8818050cb 192.168.1.115:7115 slave d2c6c159b07e8197e2c8d2eae8c847050159f602 0 1439139596712 5 connected
1fd90d54090925afb4087d4ef94a1710a25160d6 192.168.1.116:7116 myself,slave 4e46bd06654e8660e617f7249fa22f6fa1fdff0d 0 0 3 connected
48db78bcc55c4c3a3788940a6458b921ccf95d44 192.168.1.114:7114 master - 0 1439139599717 11 connected 0-165 5461-5627 10923-16383
8dd55e9b4da9f62b9b15232e86553f1337864179 192.168.1.111:7111 slave 48db78bcc55c4c3a3788940a6458b921ccf95d44 0 1439139593706 11 connected
d2c6c159b07e8197e2c8d2eae8c847050159f602 192.168.1.112:7112 master - 0 1439139597713 5 connected 5628-10922
4e46bd06654e8660e617f7249fa22f6fa1fdff0d 192.168.1.113:7113 master - 0 1439139598716 6 connected 166-5460
127.0.0.1:7116>
```