# Redis 3.2.5集群安装与测试

## 下载和解包

1. cd /usr/software
2. wget http:*//download.redis.io/releases/redis-3.2.6.tar.gz*
3. tar -zxvf /redis-3.2.6.tar.gz

## 编译安装

1. cd redis-3.2.6
2. make && make install

## 创建redis节点

测试我们选择2台服务器，分别为：192.168.215.129，192.168.215.130.每台服务器有3个节点。

1. 先在192.168.215.129创建3个节点

cd /usr/software

mkdir redis\_cluster //创建集群目录

cd redis\_cluster

mkdir 7000 7001 7002 //分别代表三个节点 其对应端口 7000 7001 7002

cd ..

//创建7000节点为例，拷贝到7000目录

cp /usr/software/redis-3.2.6/redis.conf ./redis\_cluster/7000/

//拷贝到7001目录

cp /usr/software/redis-3.2.6/redis.conf ./redis\_cluster/7001/

//拷贝到7002目录

cp /usr/software/redis-3.2.6/redis.conf ./redis\_cluster/7002/

1. 分别对7000、7001、7002文件夹中的3个文件修改对应的配置

daemonize yes //redis后台运行

pidfile /var/run/redis\_7000.pid //pidfile文件对应7000,7001,7002

port 7000 //端口7000,7002,7003

cluster-enabled yes //开启集群 把注释#去掉

cluster-config-file nodes\_7000.conf //集群的配置 配置文件首次启动自动生成 7000,7001,7002

bind 192.168.215.130 //这里要绑定机器的IP

cluster-node-timeout 5000 //请求超时 设置5秒够了

appendonly yes //aof日志开启 有需要就开启，它会每次写操作都记录一条日志

1. 在192.168.1.238创建3个节点：对应的端口改为7003,7004,7005。配置对应改一下就可以了。
2. 两台机器启动各节点（两台服务器方式一样）

cd /usr/software

redis-server redis\_cluster/7000/redis.conf

redis-server redis\_cluster/7001/redis.conf

redis-server redis\_cluster/7002/redis.conf

redis-server redis\_cluster/7003/redis.conf

redis-server redis\_cluster/7004/redis.conf

redis-server redis\_cluster/7005/redis.conf

查看服务

1. ps -ef | grep redis *#查看是否启动成功*
2. netstat -tnlp | grep redis *#可以看到redis监听端口*

[root@zk1 software]# ps -ef | grep redis

root 10601 1 0 06:42 ? 00:00:00 redis-server 192.168.215.129:7000 [cluster]

1. root 10606 1 0 06:42 ? 00:00:00 redis-server 192.168.215.129:7001 [cluster]
2. root 10610 1 0 06:42 ? 00:00:00 redis-server 192.168.215.129:7002 [cluster]
3. root 10615 4548 0 06:42 pts/2 00:00:00 grep --color=auto redis
4. [root@zk1 software]*#*
5. [root@zk1 software]*#*
6. [root@zk1 software]*#*
7. [root@zk1 software]*# netstat -tnlp | grep redis*
8. tcp 0 0 192.168.215.129:17002 0.0.0.0:\* LISTEN 10610/redis-server
9. tcp 0 0 192.168.215.129:7000 0.0.0.0:\* LISTEN 10601/redis-server
10. tcp 0 0 192.168.215.129:7001 0.0.0.0:\* LISTEN 10606/redis-server
11. tcp 0 0 192.168.215.129:7002 0.0.0.0:\* LISTEN 10610/redis-server
12. tcp 0 0 192.168.215.129:17000 0.0.0.0:\* LISTEN 10601/redis-server
13. tcp 0 0 192.168.215.129:17001 0.0.0.0:\* LISTEN 10606/redis-server

创建集群

前面已经准备好了搭建集群的redis节点，接下来我们要把这些节点都串连起来搭建集群。官方提供了一个工具：redis-trib.rb（/usr/local/redis-3.2.1/src/redis/trib.rb），看后缀就知道这东西不能直接执行，它是用ruby写的一个程序，所以我们还得安装ruby再用gem这个命令来安装redis接口。gem是ruby的一个工具包。

为了方便，两台机器都安装

1. yum -y install ruby ruby-devel rubygems rpm-build
2. gem install redis

注意：这里需要修改gem源，并且淘宝的gem源已经不能用了，现在要使用ruby-china：<https://gems.ruby-china.org/>

添加ruby-china源：

1. [root@zk1 software]*# gem sources --add https://gems.ruby-china.org/ --remove https://rubygems.org/*
2. [root@zk1 software]*# gem sources -l*
3. *\*\*\* CURRENT SOURCES \*\*\**
4. https:*//gems.ruby-china.org/*

redis-trib.rb

进入/usr/software/redis-3.2.5/src，运行一下redis-trib.rb

[root@zk2 src]# ./redis-trib.rb

Usage: redis-trib <command> <options> <arguments ...>

create host1:port1 ... hostN:portN

--replicas <arg>

check host:port

info host:port

fix host:port

--timeout <arg>

reshard host:port

--from <arg>

--to <arg>

--slots <arg>

--yes

--timeout <arg>

--pipeline <arg>

rebalance host:port

--weight <arg>

--auto-weights

--use-empty-masters

--timeout <arg>

--simulate

--pipeline <arg>

--threshold <arg>

add-node new\_host:new\_port existing\_host:existing\_port

--slave

--master-id <arg>

del-node host:port node\_id

set-timeout host:port milliseconds

call host:port command arg arg .. arg

import host:port

--from <arg>

--copy

--replace

help (show this help)

For check, fix, reshard, del-node, set-timeout you can specify the host and port of any working node in the cluster.

看到这，应该明白了吧，就是靠上面这些操作完成redis集群搭建的。

确认所有的节点都启动，接下来使用参数create创建（在192.168.215.129中来创建）

[root@zk1 src]# ./redis-trib.rb create --replicas 1 192.168.215.129:7000 192.168.215.129:7001 192.168.215.129:7002 192.168.215.130:7003 192.168.215.130:7004 192.168.215.130:7005

>>> Creating cluster

>>> Performing hash slots allocation on 6 nodes...

Using 3 masters:

192.168.215.129:7000

192.168.215.130:7003

192.168.215.129:7001

Adding replica 192.168.215.130:7004 to 192.168.215.129:7000

Adding replica 192.168.215.129:7002 to 192.168.215.130:7003

Adding replica 192.168.215.130:7005 to 192.168.215.129:7001

M: 16518afbfcbd961aeb76ef1592007a3e7fe24b1b 192.168.215.129:7000

slots:0-5460 (5461 slots) master

M: 524219969118a57ceaac753ecef7585f634cdf26 192.168.215.129:7001

slots:10923-16383 (5461 slots) master

S: ea4519ff0083a13cef8262490ee9e61e5a4b14b1 192.168.215.129:7002

replicates 82c0e591b9bc7a289026dff2873a254d1c49d285

… …

Can I set the above configuration? (type 'yes' to accept): yes

>>> Nodes configuration updated

>>> Assign a different config epoch to each node

>>> Sending CLUSTER MEET messages to join the cluster

Waiting for the cluster to join.....

>>> Performing Cluster Check (using node 192.168.215.129:7000)

M: 16518afbfcbd961aeb76ef1592007a3e7fe24b1b 192.168.215.129:7000

slots:0-5460 (5461 slots) master

1 additional replica(s)

S: ea4519ff0083a13cef8262490ee9e61e5a4b14b1 192.168.215.129:7002

slots: (0 slots) slave

replicates 82c0e591b9bc7a289026dff2873a254d1c49d285

… …

M: 524219969118a57ceaac753ecef7585f634cdf26 192.168.215.129:7001

slots:10923-16383 (5461 slots) master

1 additional replica(s)

M: 82c0e591b9bc7a289026dff2873a254d1c49d285 192.168.215.130:7003

slots:5461-10922 (5462 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.

解释下，”-replicas 1”表示自动为每一个master节点分配一个slave节点。上面有6个节点，程序会按照一定规则生成3个master、3个slave。

前面已经提醒过的，防火墙一定要开放监听的端口，否则会创建失败。

检查集群状态：

[root@zk1 src]# ./redis-trib.rb check 192.168.215.129:7002

>>> Performing Cluster Check (using node 192.168.215.129:7002)

S: ea4519ff0083a13cef8262490ee9e61e5a4b14b1 192.168.215.129:7002

slots: (0 slots) slave

replicates 82c0e591b9bc7a289026dff2873a254d1c49d285

M: 82c0e591b9bc7a289026dff2873a254d1c49d285 192.168.215.130:7003

slots:5461-10922 (5462 slots) master

1 additional replica(s)

S: baf74dd89c0605d2a71a8d1d3706005ff668563b 192.168.215.130:7004

slots: (0 slots) slave

replicates 16518afbfcbd961aeb76ef1592007a3e7fe24b1b

M: 524219969118a57ceaac753ecef7585f634cdf26 192.168.215.129:7001

slots:10923-16383 (5461 slots) master

1 additional replica(s)

M: 16518afbfcbd961aeb76ef1592007a3e7fe24b1b 192.168.215.129:7000

slots:0-5460 (5461 slots) master

1 additional replica(s)

S: f8192314d2232e12ba9f558e9ecbfcc890f4fb73 192.168.215.130:7005

slots: (0 slots) slave

replicates 524219969118a57ceaac753ecef7585f634cdf26

[OK] All nodes agree about slots configuration.

>>> Check for open slots...

>>> Check slots coverage...

[OK] All 16384 slots covered.

测试集群

* 连接服务器

[root@zk1 src]# ./redis-cli -c -p 7000 -h 192.168.215.129

192.168.215.129:7000>

192.168.215.129:7000> cluster info

cluster\_state:ok

cluster\_slots\_assigned:16384

cluster\_slots\_ok:16384

cluster\_slots\_pfail:0

cluster\_slots\_fail:0

cluster\_known\_nodes:6

cluster\_size:3

cluster\_current\_epoch:6

cluster\_my\_epoch:1

cluster\_stats\_messages\_sent:1502

cluster\_stats\_messages\_received:1502

* set值

1. 192.168.215.129:7000> set name lbl
2. -> Redirected to slot [5798] located at 192.168.215.130:7003
3. OK
4. 192.168.215.130:7003> get name
5. "lbl"

可见，重定向到了130节点7003端口。

原因是redis采用hash槽的方式分发key到不同节点，算法是crc(16)%16384。详细描述后续会单独写文章描述。

而且你会发现，当一次重定向以后，这个客户端就连接到了130:7003这个节点。

测试一个master宕机

将上面设置的name所在的130:7003 kill掉，只剩了两个master和3个slave。你会发现cluster\_current\_epoch相比之前加了1，这是因为redis的主从关系，重新选了一次主，然后get name发现，重定向了129:7002这个节点。

[root@zk1 src]# ./redis-trib.rb check 192.168.215.129:7002

>>> Performing Cluster Check (using node 192.168.215.129:7002)

M: ea4519ff0083a13cef8262490ee9e61e5a4b14b1 192.168.215.129:7002

slots:5461-10922 (5462 slots) master

0 additional replica(s)

S: baf74dd89c0605d2a71a8d1d3706005ff668563b 192.168.215.130:7004

slots: (0 slots) slave

replicates 16518afbfcbd961aeb76ef1592007a3e7fe24b1b

M: 524219969118a57ceaac753ecef7585f634cdf26 192.168.215.129:7001

slots:10923-16383 (5461 slots) master

1 additional replica(s)

M: 16518afbfcbd961aeb76ef1592007a3e7fe24b1b 192.168.215.129:7000

slots:0-5460 (5461 slots) master

1 additional replica(s)

S: f8192314d2232e12ba9f558e9ecbfcc890f4fb73 192.168.215.130:7005

slots: (0 slots) slave

replicates 524219969118a57ceaac753ecef7585f634cdf26

[OK] All nodes agree about slots configuration.

>>> Check for open slots...

>>> Check slots coverage...

[OK] All 16384 slots covered.

[root@zk1 src]# ./redis-cli -c -p 7000 -h 192.168.215.129

192.168.215.129:7000> cluster info

cluster\_state:ok

cluster\_slots\_assigned:16384

cluster\_slots\_ok:16384

cluster\_slots\_pfail:0

cluster\_slots\_fail:0

cluster\_known\_nodes:6

cluster\_size:3

cluster\_current\_epoch:7

cluster\_my\_epoch:1

cluster\_stats\_messages\_sent:2883

cluster\_stats\_messages\_received:2675

192.168.215.129:7000> get name

-> Redirected to slot [5798] located at 192.168.215.129:7002

"lbl"

之所以会重定向到129:7002这个节点，是因为在kill之前，129:7002是130:7003的slave。下面这是在kill之前拷贝的”./redis-trib.rb check”的数据，注意replicates后的值：

1. `S: ea4519ff0083a13cef8262490ee9e61e5a4b14b1 192.168.215.129:7002`

2. `replicates 82c0e591b9bc7a289026dff2873a254d1c49d285`

3. `M: 82c0e591b9bc7a289026dff2873a254d1c49d285 192.168.215.130:7003`

4. `slots:5461-10922 (5462 slots) master`

redis具体实现原理后续再讲。