Linux环境下Redis Cluster 安装部署手册

# 1．Redis集群简介

Redis Cluster是Redis官方的分布式解决方案，在3.0版本正式推出，有效地解决了Redis分布式方面的需求。当遇到单机内存、并发、流量等瓶颈时，可

以采用Cluster架构方案达到负载均衡的目的。Redis集群一般由多个节点组成，节点数量至少为6个才能保证组成完整高可用的集群。每个节点需要开启配置cluster-enabled yes，让Redis运行在集群模式下。

## 部署环境

3台 Ubuntu 16.04 Linux虚拟机，物理IP分别为：192.168.15.1 、192.168.15.2 、192.168.15.3 。

## 1.2 Redis版本

V3.2.11

## 集群方案

|  |  |  |  |
| --- | --- | --- | --- |
| 主机IP | Redis服务端口 | Master/Slave | 所属Master |
| 192.168.15.1 | 7000 | M | - |
| 192.168.15.1 | 7001 | S | 192.168.15.2:7003 |
| 192.168.15.1 | 7002 | S | 192.168.15.3:7006 |
|  | | | |
| 192.168.15.2 | 7003 | M | - |
| 192.168.15.2 | 7004 | S | 192.168.15.1:7000 |
| 192.168.15.2 | 7005 | S | 192.168.15.3:7006 |
|  | | | |
| 192.168.15.3 | 7006 | M | - |
| 192.168.15.3 | 7007 | S | 192.168.15.1:7000 |
| 192.168.15.3 | 7008 | S | 192.168.15.2:7003 |

# 2．安装Redis

## 2.1 下载Redis安装包

在线下载

$ wget <http://redis.googlecode.com/files/redis-3.2.11.tar.gz>

或者将下载好的redis-3.2.11.tar.gz安装包上传到Linux服务器指定目录下，比如/usr/local/。

## 2.2 解压

进入redis-3.2.11.tar.gz所在目录，命令行输入如下命令：

$ tar –zxvf redis-3.2.11.tar.gz

## 2.3 编译安装

### 2.3.1 编译Redis

进入Redis解压后目录

$ cd redis-3.2.11.tar.gz

编译

$ make

如果未成功，显示如下提示：The program ‘make’ can be found in the following packages:等主要是因为没有安装GCC环境及make工具，请执行步骤2.3.2、2.3.3步骤后再make 。

### 2.3.2 安装GCC

执行如下命令

$ sudo apt-get install gcc

确认

$ Y

### 2.3.3 安装make程序

在命令行输入如下命令，完成make工具安装

$ sudo apt-get install build-essential

确认

$ Y

### 2.3.4 make检查

$ cd src && make test

如果提示：You need tcl 8.5 or newer in order to run the Redis test ，请执行步骤2.3.5完成tcl工具安装（tcl工具不安装并不影响Redis使用，为了安装环境完备性，建议安装）后，再执行此检查。

如跑完之后，有如下提示，表明安装成功。

\o/ All tests passed without errors!

### 2.3.5 安装Tcl

1）下载安装包

$ wget http://downloads.sourceforge.net/tcl/tcl8.6.1-src.tar.gz

2）解压

$ sudo tar -zxvf tcl8.6.1-src.tar.gz

3）进入目录

$ cd tcl8.6.1/unix

4）编译安装

$ sudo ./configure

$ sudo make

$ sudo make install

### 2.3.6 安装Redis

可以执行如下命令，可以Redis安装，并将Redis的二进制文件生成到/usr/local/bin目录下。

$ sudo make install

或者执行如下命令，生成到自定义的目的目录。

$ sudo make install PREFIX=/usr/local/redis

### 2.3.7 init脚本

$ sudo cd utils # 进入 redis-3.2.11目录下的utils目录下

$ sudo ./install\_server.sh

过程中会询问配置信息，可以按照默认执行。

运行该命令后，可以通过/etc/init.d/redis\_6379 stop/start/restart来快速关闭、启动、重启Redis服务。

## 2.4 测试Redis

### 2.4.1 Redis服务端启动

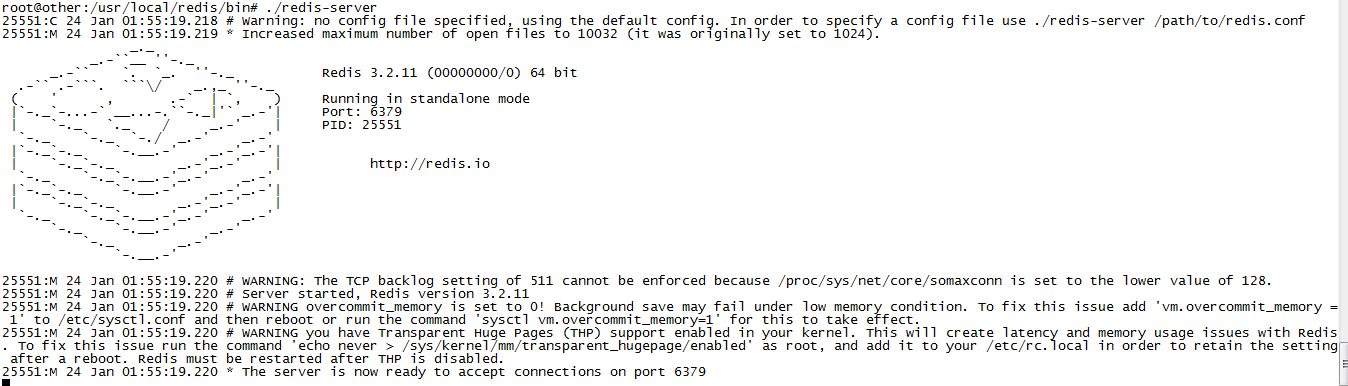
1）按照默认配置，启动（6379端口）

$ ./redis-server

2）指定启动配置文件启动

$ ./redis-server /path/to/redis.conf

显示如下图标后，表示Redis启动成功。（daemonize yes时，下图不显示）



### 2.4.2 Redis客户端连接

新开一个终端，进入redis-cli命令所在目录，在命令行输入

$ ./redis-cli

可以开启连接默认6379的客户端

或者输入命令参数，访问指定Redis服务，如下命令表示访问IP为127.0.0.1上端口7000的Redis服务。

$ ./redis-cli -h 127.0.0.1 –p 7000



## 2.5 设置开机启动

待完善。

## 2.6 安全设置

待完善。

# 3．搭建Redis集群

## 3.1 集群工具准备

Redis官方提供了集群快速搭建的脚本工具redis-trib.rb（Redis安装后的src目录下），该脚本工具是由Ruby编写的，需要安装如下依赖环境。

### 3.1.1 安装Ruby

$ sudo apt-get update

$ sudo apt-get install ruby

安装完成后，可以在控制台输入 #ruby -v ，如能显示ruby版本信息，表明Ruby安装成功。

### 3.1.2 安装Redis Ruby依赖

1）安装RubyGems

$ sudo apt-get install rubygems

2）更新RubyGems

$ sudo gem install rubygems-update

3）在Redis目录下执行命令：

$ sudo gem install redis

## 3.2 准备节点

### 3.2.1 创建节点统一目录

确保各服务器已经成功安装Redis，如未安装请按章节2步骤完成Redis安装。

本例中192.168.15.1 、192.168.15.2、192.168.15.3 镜像安装Redis，故可以采用/usr/local/redis目录作为各主机节点的统一目录。

### 3.2.2 创建节点目录

在节点统一目录下，创建节点目录，以端口命名

1. 在主机192.168.15.1上操作如下

$ cd /usr/local/redis

$ sudo mkdir 7000 7001 7002

1. 在主机192.168.15.2上操作如下

$ cd /usr/local/redis

$ sudo mkdir 7003 7004 7005

1. 在主机192.168.15.3上操作如下

$ cd /usr/local/redis

$ sudo mkdir 7006 7007 7008

### 3.2.3 复制redis.conf配置文件至每个节点目录

可从Redis解压安装目录下复制redis.conf文件

### 修改每个节点的redis.cinf配置文件

以192.168.15.1主机上 7000节点为例。

$ cd /usr/local/redis/7000

$ vi redis.conf

配置文件做如下修改

port 7000

daemonize yes

loglevel notice

cluster-enabled yes

cluster-config-file nodes.7000.conf

cluster-node-timeout 15000

cluster-migration-barrier 1

cluster-require-full-coverage yes

appendonly yes

appendfilename "appendonly.7000.aof"

其他Redis实例参照修改为自己的端口号等信息。

## 启动节点

### 3.3.1 以各端口配置文件启动Redis节点服务

本例中启动7000端口的Redis服务实例步骤如下

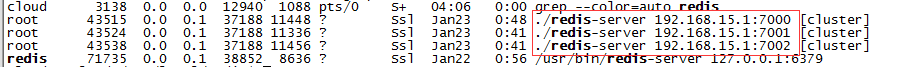
$ cd /usr/local/redis/bin

$ ./redis-server ../7000/redis-conf

参照启动 192.168.15.1上的7001、7002 节点服务，并启动192.168.15.2上7003、7004、7005 以及192.168.15.2上7006、7007、7008节点服务。

### 3.3.2 检查Redis节点服务是否启动

$ ps –aux |grep redis



## 创建集群

进入redis-trib.rb文件所在目录执行：

$ ruby redis-trib.rb create --replicas 2 192.168.15.1:7000 192.168.15.2:7003

192.168.15.3:7006 192.168.15.2:7004 192.168.15.3:7007 192.168.15.1:7001 192.168.15.3:7008 192.168.15.1:7002 192.168.15.2:7005

该命令用于创建一个新的集群, 选项--replicas 2 表示我们希望为集群中的每个主节点创建2个从节点。之后跟着的其他参数则是这个集群实例的地址列表,3个master 、6个slave。

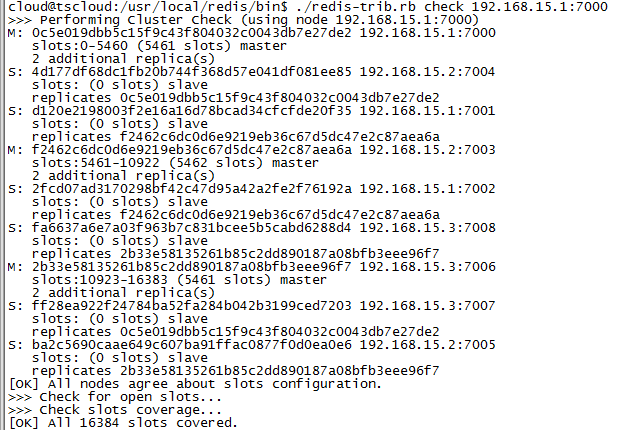
redis-trib 会打印出一份预想中的配置给你看， 如果你觉得没问题的话， 就可以输入 yes ，redis-trib 就会将这份配置应用到集群当中,让各个节点开始互相通讯。

## 检验集群

$ ./redis-trib.rb check 192.168.15.1:7000

$ ./redis-cli -c -h 192.168.15.1 -p 7000 cluster nodes

部分集群信息如下：



## 访问集群

$ ./redis-cli -c –h 192.168.15.1 -p 7000

# 附录A：Redis配置文件说明

Redis配置文件redis.conf 共有60多项配置项，现介绍其中常用的几个配置参数项目。

1. Redis默认不是以守护进程的方式运行，可以通过该配置项修改，使用yes启用守护进程

daemonize no

2. 当Redis以守护进程方式运行时，Redis默认会把pid写入/var/run/redis.pid文件，可以通过pidfile指定

pidfile /var/run/redis.pid

3. 指定Redis监听端口，默认端口为6379，作者在自己的一篇博文中解释了为什么选用6379作为默认端口，因为6379在手机按键上MERZ对应的号码，而MERZ取自意大利歌女Alessia Merz的名字

port 6379

4. 绑定的主机地址

bind 127.0.0.1

5.当客户端闲置多长时间后关闭连接，如果指定为0，表示关闭该功能

timeout 300

6. 指定日志记录级别，Redis总共支持四个级别：debug、verbose、notice、warning，默认为verbose

loglevel notice

7. 日志记录方式，默认为标准输出，如果配置Redis为守护进程方式运行，而这里又配置为日志记录方式为标准输出，则日志将会发送给/dev/null

logfile stdout

8. 设置数据库的数量，默认数据库为0，可以使用SELECT <dbid>命令在连接上指定数据库id

databases 16

9. 指定在多长时间内，有多少次更新操作，就将数据同步到数据文件，可以多个条件配合

save <seconds> <changes>

Redis默认配置文件中提供了三个条件：

save 900 1

save 300 10

save 60 10000

分别表示900秒（15分钟）内有1个更改，300秒（5分钟）内有10个更改以及60秒内有10000个更改。

10. 指定存储至本地数据库时是否压缩数据，默认为yes，Redis采用LZF压缩，如果为了节省CPU时间，可以关闭该选项，但会导致数据库文件变的巨大

rdbcompression yes

11. 指定本地数据库文件名，默认值为dump.rdb

dbfilename dump.rdb

12. 指定本地数据库存放目录

dir ./

13. 设置当本机为slave服务时，设置master服务的IP地址及端口，在Redis启动时，它会自动从master进行数据同步

slaveof <masterip> <masterport>

14. 当master服务设置了密码保护时，slav服务连接master的密码

masterauth <master-password>

15. 设置Redis连接密码，如果配置了连接密码，客户端在连接Redis时需要通过AUTH <password>命令提供密码，默认关闭

requirepass foobared

16. 设置同一时间最大客户端连接数，默认无限制，Redis可以同时打开的客户端连接数为Redis进程可以打开的最大文件描述符数，如果设置 maxclients 0，表示不作限制。当客户端连接数到达限制时，Redis会关闭新的连接并向客户端返回max number of clients reached错误信息

maxclients 128

17. 指定Redis最大内存限制，Redis在启动时会把数据加载到内存中，达到最大内存后，Redis会先尝试清除已到期或即将到期的Key，当此方法处理后，仍然到达最大内存设置，将无法再进行写入操作，但仍然可以进行读取操作。Redis新的vm机制，会把Key存放内存，Value会存放在swap区

maxmemory <bytes>

18. 指定是否在每次更新操作后进行日志记录，Redis在默认情况下是异步的把数据写入磁盘，如果不开启，可能会在断电时导致一段时间内的数据丢失。因为 redis本身同步数据文件是按上面save条件来同步的，所以有的数据会在一段时间内只存在于内存中。默认为no ，即Redis默认是使用快照方式实现持久化。

appendonly yes

19. 指定更新日志文件名，默认为appendonly.aof

appendfilename appendonly.aof

20. 指定更新日志条件，共有3个可选值：

no：表示等操作系统进行数据缓存同步到磁盘（快）

always：表示每次更新操作后手动调用fsync()将数据写到磁盘（慢，安全）

everysec：表示每秒同步一次（折衷，默认值）

appendfsync everysec

21. 指定是否启用虚拟内存机制，默认值为no，简单的介绍一下，VM机制将数据分页存放，由Redis将访问量较少的页即冷数据swap到磁盘上，访问多的页面由磁盘自动换出到内存中（在后面的文章我会仔细分析Redis的VM机制）

vm-enabled no

22. 虚拟内存文件路径，默认值为/tmp/redis.swap，不可多个Redis实例共享

vm-swap-file /tmp/redis.swap

23. 将所有大于vm-max-memory的数据存入虚拟内存,无论vm-max-memory设置多小,所有索引数据都是内存存储的(Redis的索引数据 就是keys),也就是说,当vm-max-memory设置为0的时候,其实是所有value都存在于磁盘。默认值为0

vm-max-memory 0

24. Redis swap文件分成了很多的page，一个对象可以保存在多个page上面，但一个page上不能被多个对象共享，vm-page-size是要根据存储的 数据大小来设定的，作者建议如果存储很多小对象，page大小最好设置为32或者64bytes；如果存储很大大对象，则可以使用更大的page，如果不 确定，就使用默认值

vm-page-size 32

25. 设置swap文件中的page数量，由于页表（一种表示页面空闲或使用的bitmap）是在放在内存中的，，在磁盘上每8个pages将消耗1byte的内存。

vm-pages 134217728

26. 设置访问swap文件的线程数,最好不要超过机器的核数,如果设置为0,那么所有对swap文件的操作都是串行的，可能会造成比较长时间的延迟。默认值为4

vm-max-threads 4

27. 设置在向客户端应答时，是否把较小的包合并为一个包发送，默认为开启

glueoutputbuf yes

28. 指定在超过一定的数量或者最大的元素超过某一临界值时，采用一种特殊的哈希算法

hash-max-zipmap-entries 64

hash-max-zipmap-value 512

29. 指定是否激活重置哈希，默认为开启（后面在介绍Redis的哈希算法时具体介绍）

activerehashing yes

30. 指定包含其它的配置文件，可以在同一主机上多个Redis实例之间使用同一份配置文件，而同时各个实例又拥有自己的特定配置文件

include /path/to/local.conf

31.集群开关，默认是不开启集群模式

cluster-enabled yes

32.设置集群配置文件，集群模式下，每个redis节点生成一个自己的集群配置文件，这个文件不需要人工修改，由redis自己维护

cluster-config-file nodes.7000.conf

33. 集群模式时，当前节点在与其他节点保活探测时，多久没有响应时认为其他节点处于fail状态，以下表示是15秒

cluster-node-timeout 15000

34. 是否开启appendonlylog，开启的话每次写操作会记一条log，这会提高数据抗风险能力，但影响效率

appendonly yes

35.aof文件名

appendfilename "appendonly.7000.aof"

36. 默认情况下，集群全部的slot有节点负责，集群状态才为ok，才能提供服务。

设置为no，可以在slot没有全部分配的时候提供服务。不建议打开该配置，这样会造成分区的时候，小分区的master一直在接受写请求，而造成很长时间数据不一致。默认为Yes,丢失一定比例Key后（可能Node无法连接或者挂掉），集群停止接受写操作 no:允许某些slot没有active节点的时候其余slot还能用。就是其中1个实例挂掉，不会导致整个redis cluster都不可用了。

cluster-require-full-coverage yes

37. 控制从节点FailOver相关的设置。设为0，从节点会一直尝试启动FailOver.设为正数，失联大于一定时间（factor\*节点TimeOut），不再进行FailOver)

cluster-slave-validity-factor 10

38.保护模式，默认是开启状态，只允许本地客户端连接， 可以设置密码或添加bind来连接

protected-mode yes