# 搭建redis集群

**一、安装需要的依赖包**

[root@localhost ~]# yum install gcc gcc-c++ kernel-devel automake autoconf libtool make wget tcl vim ruby rubygems unzip git -y

**二、下载redis并安装**

1.安装redis

[root@localhost src]# cd /usr/local/

[root@localhost local]# wget <http://download.redis.io/releases/redis-3.2.8.tar.gz>

[root@localhost local]# tar xzf redis-3.2.8.tar.gz

[root@localhost local]# cd redis-3.2.8

[root@localhost redis-3.2.8]# make && make install

2.安装redis和ruby的接口

联网状态下：# gem install redis

离线本地安装：<https://rubygems.org/gems/redis/versions/3.2.2>

#gem install --local redis-3.2.2.gem

**三、修改配置文件.conf**

1)创建配置文件目录，dump file 目录，进程pid目录，log目录等

配置文件一般放在/etc/下，创建redis目录

cd /etc/

mkdir redis

http://img.blog.csdn.net/20150804101536363

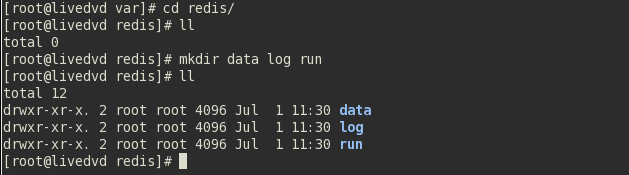
ll 查看创建的redis目录、dump file、进程pid、log目录等，一般放/var/目录下，

cd /var/

mkdir redis

cd redis

mkdir data log run

至此，目录创建完毕

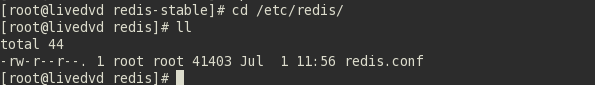
2)修改配置文件，配置参数

首先拷贝解压包下的redis.conf文件至/etc/redis

http://img.blog.csdn.net/20150804101630361

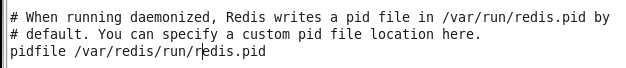
查看/etc/redis/redis.conf

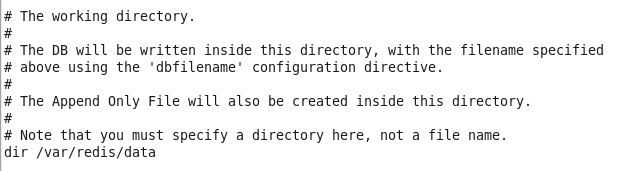
cd /etc/redis/

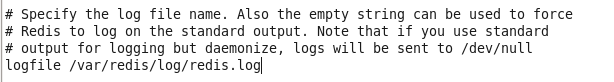
ll

打开redis.conf文件

修改端口(默认6379)http://img.blog.csdn.net/20150804101732828

修改pid目录为新建目录

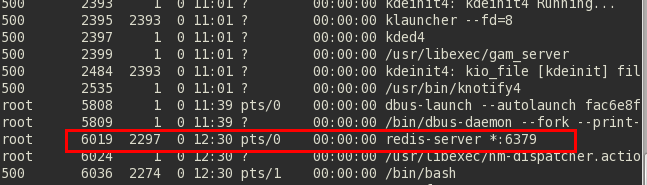
修改dump目录为新建目录

修改log存储目录为新建目录

3)持久化

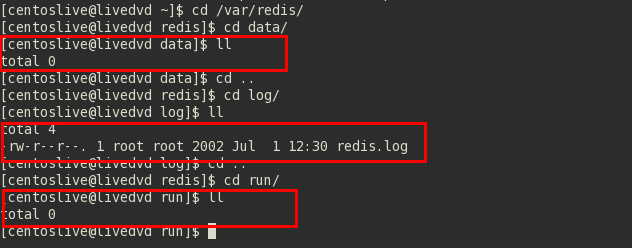
默认rdb，可选择是否开启aof，若开启，修改配置文件appendonly

4)启动redis，查看各目录下文件http://img.blog.csdn.net/20150804102017859

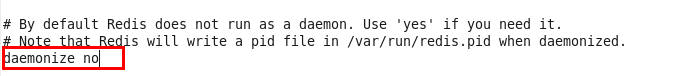
查看进程

redis已启动

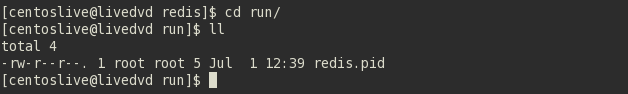
查看dump, log, pid等

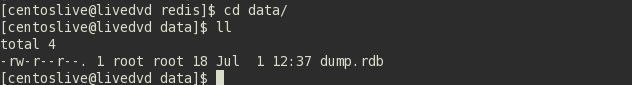


发现只有日志，没有dump和pid信息，是因为当前redis服务仍然是console模式运行的，且没有数据存储操作

停止redis服务，修改配置文件使得redis在background运行

改成yes，保存，重启redis服务

查看pid信息，如下

查看dump信息

若配置了aof持久化方式，data目录下还会有aof的相关文件

5)客户端连接redis

http://img.blog.csdn.net/20150804102239065

默认端口6379

**四、创建集群前的配置**

mkdir -p /usr/local/cluster  
cd /usr/local/cluster  
mkdir 7006  
mkdir 7001  
mkdir 7002

mkdir 7003  
mkdir 7004  
mkdir 7005

修改配置文件redis.conf  
cp /usr/local/redis-3.2.8/redis.conf  /usr/local/cluster  
cd /usr/local/cluster  
vi redis.conf

##注意每个实例的端口号不同  
port 7006  
daemonize yes  
cluster-enabled yes  
cluster-config-file nodes.conf  
cluster-node-timeout 5000  
appendonly yes

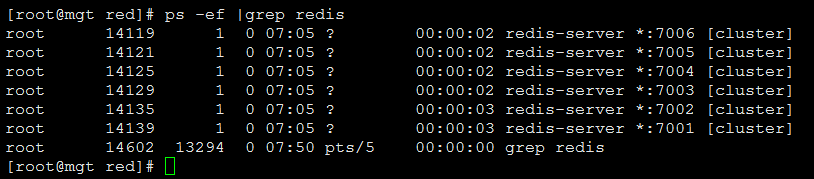
##修改完redis.conf配置文件中的这些配置项之后把这个配置文件分别拷贝到7006/7001/7002/7003/7004/7005节点目录下  
server1执行：

cp /usr/local/cluster/redis.conf /usr/local/cluster/7006  
cp /usr/local/cluster/redis.conf /usr/local/cluster/7001  
cp /usr/local/cluster/redis.conf /usr/local/cluster/7002

cp /usr/local/cluster/redis.conf /usr/local/cluster/7003  
cp /usr/local/cluster/redis.conf /usr/local/cluster/7004  
cp /usr/local/cluster/redis.conf /usr/local/cluster/7005

##注意：拷贝完成之后要分别修改7001/7002/7003/7004/7005目录下面redis.conf文件中的port参数，分别改为对应的文件夹的名称

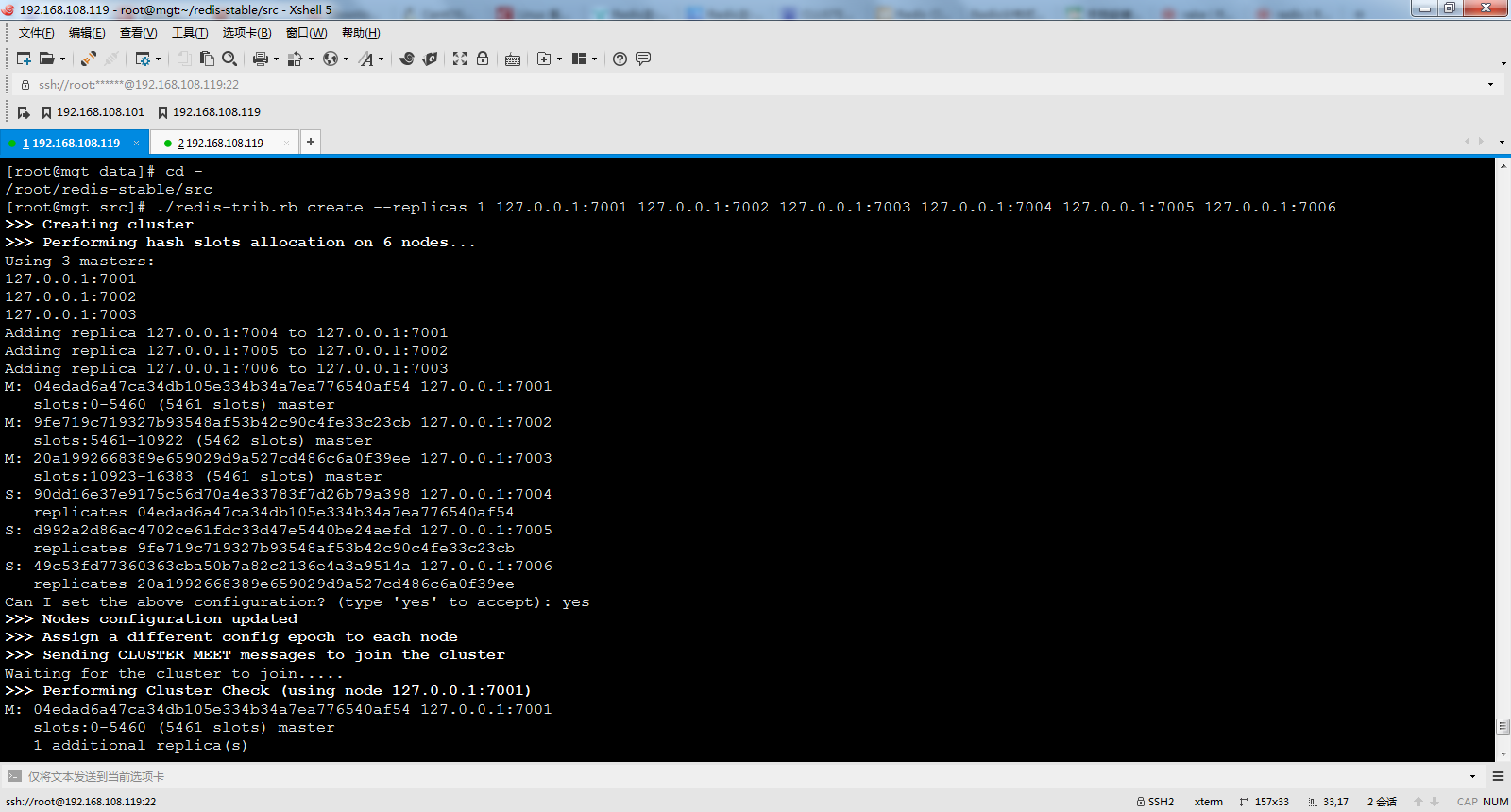
**五、启动实例**

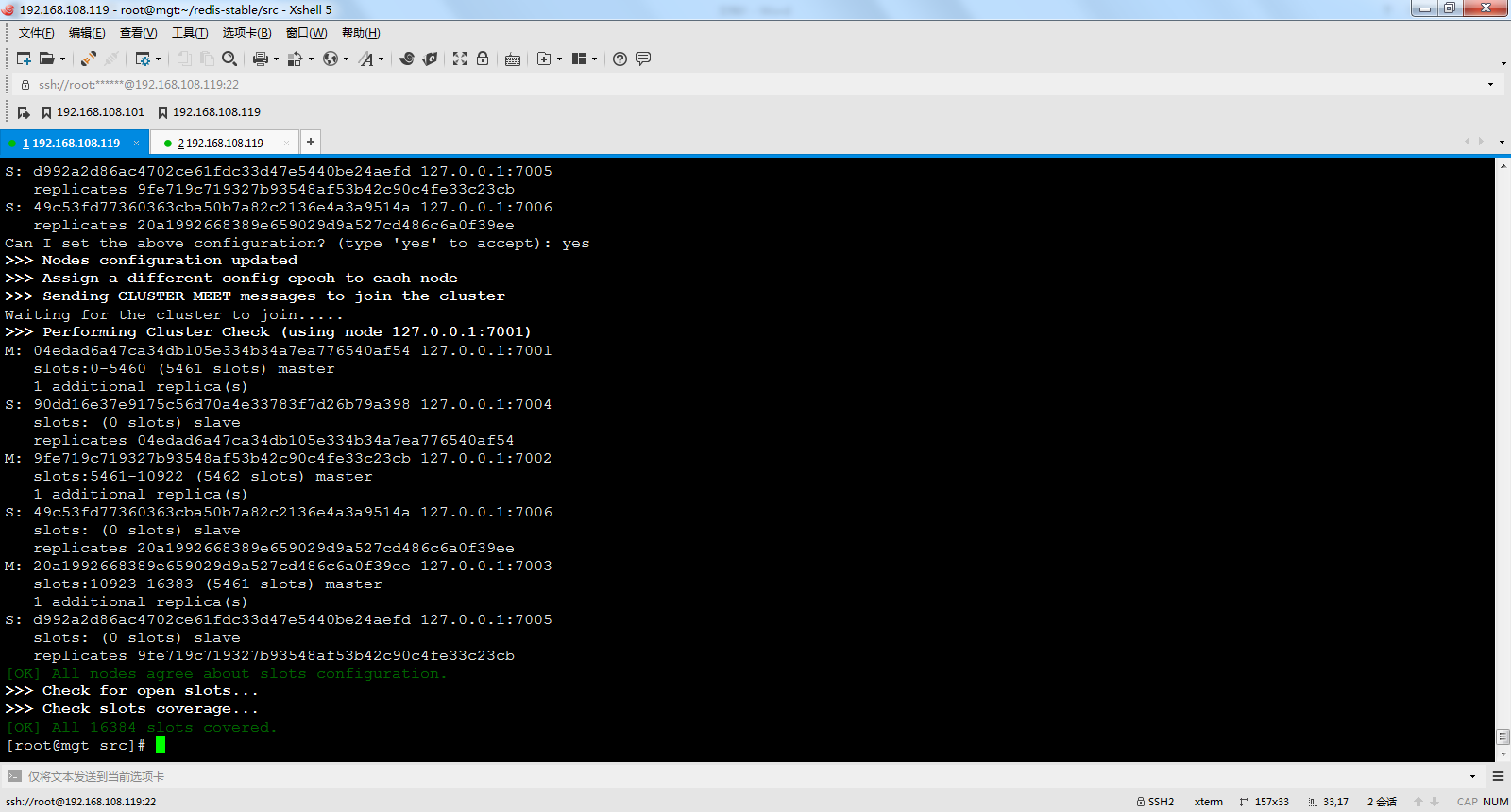
分别启动这6个redis实例，并查看是否成功：ps -ef|grep redis  
[root@localhost cluster]# cd /usr/local/cluster/7000  
[root@localhost 7006]# redis-server redis.conf  
[root@localhost 7000]# cd /usr/local/cluster/7001  
[root@localhost 7001]# redis-server redis.conf  
[root@localhost 7001]# cd /usr/local/cluster/7002  
[root@localhost 7002]# redis-server redis.conf  
[root@localhost cluster]# cd /usr/local/cluster/7003  
[root@localhost 7003]# redis-server redis.conf  
[root@localhost 7003]# cd /usr/local/cluster/7004  
[root@localhost 7004]# redis-server redis.conf  
[root@localhost 7004]# cd /usr/local/cluster/7005  
[root@localhost 7005]# redis-server redis.conf  
[root@localhost 7005]# ps -ef|grep redis  


**六、创建集群**

# cd /usr/local/redis-3.0.6/src

./redis-trib.rb create --replicas 1 127.0.0.1:7001 127.0.0.1:7002 127.0.0.1:7003 127.0.0.1:7004 127.0.0.1:7005 127.0.0.1:7006

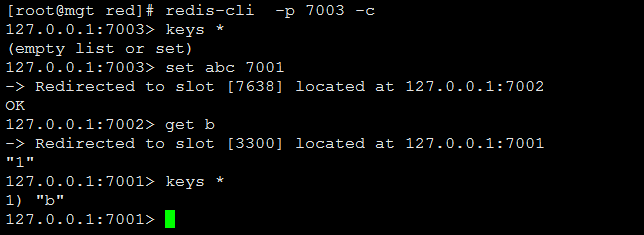




**七、测试集群**

1）使用任意一个节点客户端登陆 主要加上 -c参数表示用于集群连接

2）任意执行一个set命令，节点间会自动转发，将set的值保存到对应的节点slot中。



**八、redis cluster 架构**

1)redis-cluster架构图



架构细节:

(1)所有的redis节点彼此互联(PING-PONG机制),内部使用二进制协议优化传输速度和带宽.

(2)节点的fail是通过集群中超过半数的节点检测失效时才生效.

(3)客户端与redis节点直连,不需要中间proxy层.客户端不需要连接集群所有节点,连接集群中任何一个可用节点即可

(4)redis-cluster把所有的物理节点映射到[0-16383]slot上,cluster 负责维护node<->slot<->value

2) redis-cluster选举:容错



(1)领着选举过程是集群中所有master参与,如果半数以上master节点与master节点通信超过(cluster-node-timeout),认为当前master节点挂掉.

(2):什么时候整个集群不可用(cluster\_state:fail),当集群不可用时,所有对集群的操作做都不可用，收到((error) CLUSTERDOWN The cluster is down)错误

    a:如果集群任意master挂掉,且当前master没有slave.集群进入fail状态,也可以理解成进群的slot映射[0-16383]不完成时进入fail状态.

    b:如果进群超过半数以上master挂掉，无论是否有slave集群进入fail状态.