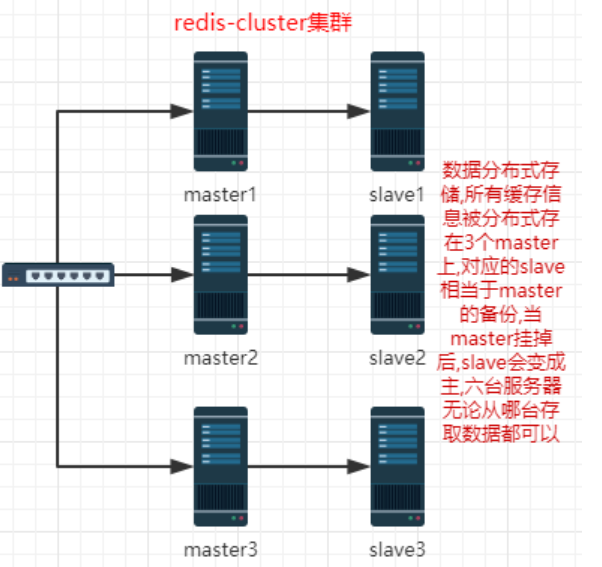
### Redis Cluster



官方参考文档 <https://redis.io/topics/cluster-tutorial>



## Redis集群

 Redis集群是一个可以在多个Redis节点之间进行数据共享的设施( installation )。

Redis集群不支持那些需要同时处理多个键的Redis命令，因为执行这些命令需要在多个Redis节点之间移动数据，并且在高负载的情况下，这些命令将降低Redis集群的性能，并导致不可预测的行为。

Redis集群通过分区( partition )来提供一定程度的可用性( availability ) : 即使集群中有一部分节点失效或者无法进行通讯，集群也可以继续处理命令请求。

将数据自动切分( split) 到多个节点的能力。

当集群中的一部分节点失效或者无法进行通讯时，仍然可以继续处理命令请求的能力。

## Redis集群数据共享

Redis集群使用数据分片( sharding )而非一致性哈希(consistency hashing )来实现:一个Redis集群包含16384个哈希槽( hash slot )， 数据库中的每个键都属于这16384个哈希槽的其中一个， 集群使用公式CRC16(key) % 16384来计算键key属于哪个槽，其中CRC16(key)语句用于计算键key的CRC16校验和。  
节点A负责处理0号至5500号哈希槽。  
节点B负责处理5501号至11000号哈希槽。  
节点C负责处理11001号至16384号哈希槽。

## 集群的复制

为了使得集群在一部分节点下线或者无法与集群的大多数( majority )节点进行通讯的情况下，仍然可以正常运作 ，Redis 集群对节点使用了主从复制功能:集群中的每个节点都有1个至N个复制品( replica )，其中一个复 制品为主节点( master ), 而其余的N-1个复制品为从节点( slave)  
在之前列举的节点A、B、C的例子中，如果节点B下线了，那么集群将无法正常运行， 因为集群找不到节点来处理5501号至11000号的哈希槽。  
假如在创建集群的时候(或者至少在节点B下线之前)，我们为主节点B添加了从节点B1， 那么当主节点B下线的时候，集群就会将 B1设置为新的主节点，并让它代替下线的主节点B，继续处理5501号至11000号的哈希槽这样集群就不会因为主节点B的下线而无法正常运作了。  
不过如果节点B和B1都下线的话，Redis集群还是会停止运作。

## 运行机制

所有的redis节点彼此互联(PINGPONG机制)，内部使用二进制协议优化传输速度和带宽.  
节点的fail（失效）是通过集群中超过半数的master节点检测失效时才生效  
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  
把所有的物理节点映射到[0-16383]slot.上,cluster负责维护node<>slot<.>key

## Redis集群特性

redis cluster集群方式支持主从自动切换   
redis cluster集群只有一个库，单例或者主从的话有多个库   
redis cluster集群去中心化，只要通过其中一个端口连接即可   
redis cluster集群，只有一个db库，不支持多库

单实例的并发QBS每秒11万次请求 读写差距不大 大约1W左右  
单实例QBS不要超过5W 超过五万扩容集群 一般3W左右 7W是极限

### Building Redis Cluster

### 1.1、安装redis

wget http://download.redis.io/releases/redis-5.0.5.tar.gz

tar xzf redis-5.0.5.tar.gz –C /usr/local/

cd redis-5.0.5

make

### 1.2、配置redis

将redis.sh redis-cluster.tmpl 上传至 /home/redis-cluster目录下

redis-cluster.tmpl 配置如下：

# bind 127.0.0.1

port ${PORT}

protected-mode no

cluster-enabled yes

cluster-config-file nodes-${PORT}.conf

cluster-node-timeout 5000

appendonly yes

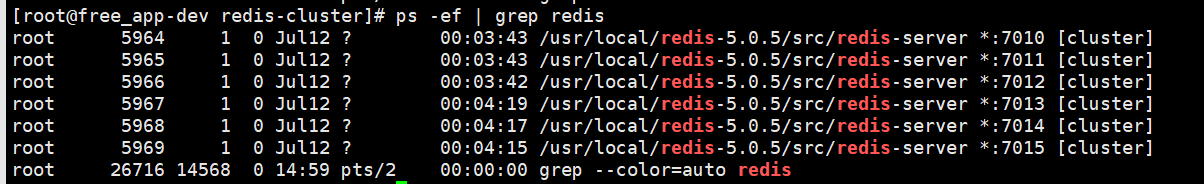
dir /home/redis-cluster/${PORT}/data

### 1.3、运行redis

1．运行脚本 redis.sh create 创建Redis Cluster 目录及配置文件。

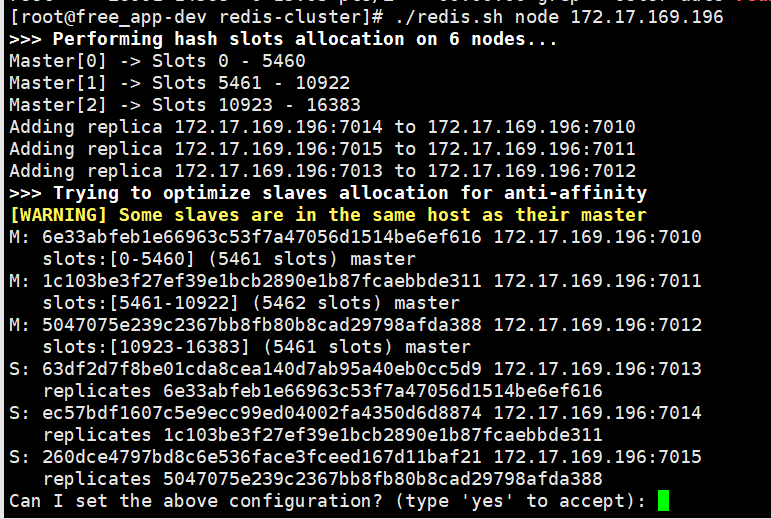
2．运行脚本 redis.sh start 创建Redis Cluster进程服务。

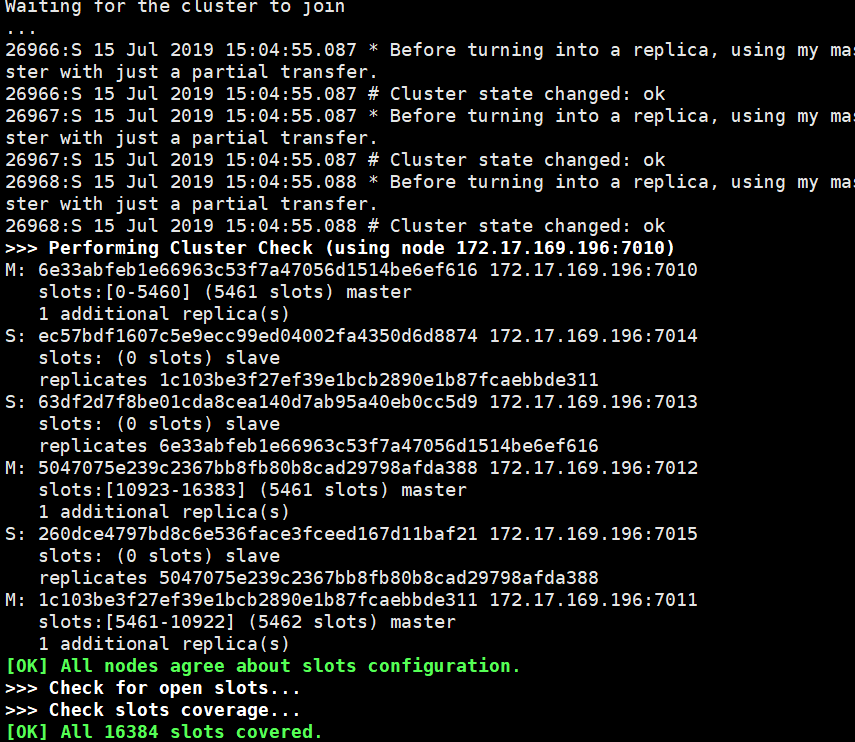
查看redis进程：



### 1.4、创建redis集群

1．运行脚本 redis.sh node + 内网IP

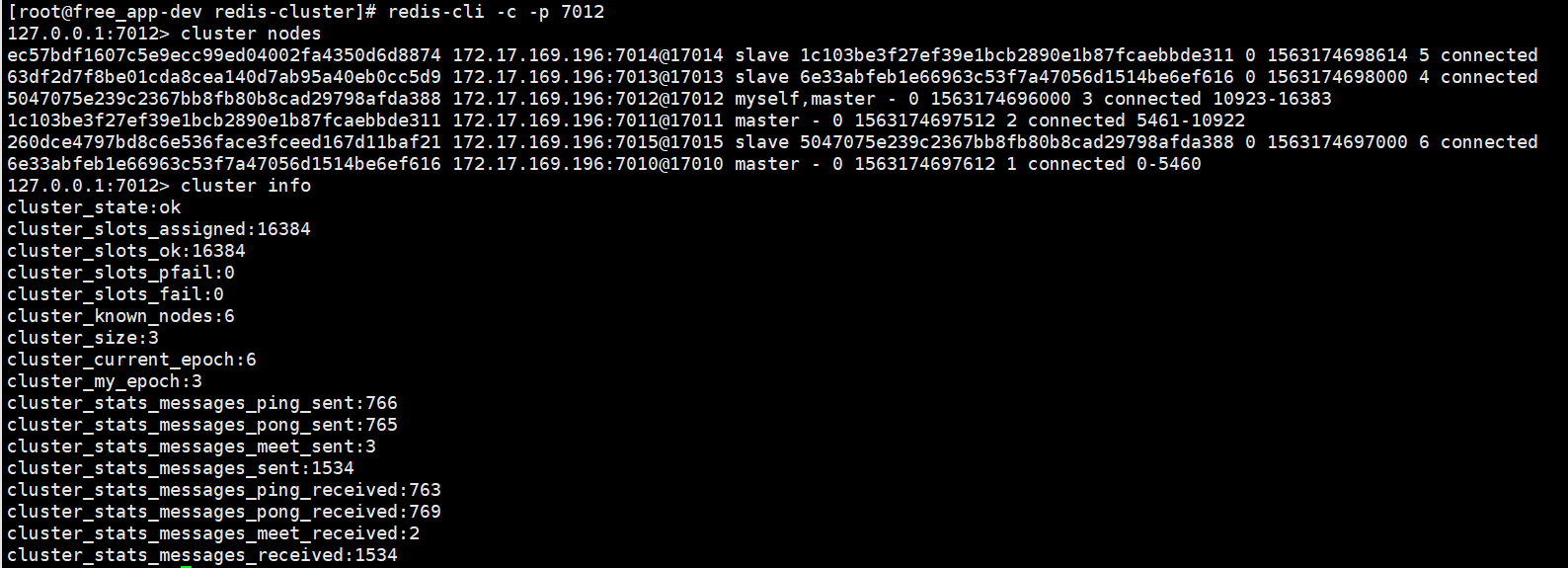




### 1.5、测试是否安装成功

[root@rojao2 ~]# redis-cli -c -h 10.10.2.208 -p 7000 cluster info

集群状态ok代表集群可用



### 1.6、重启redis

停止redis：

./redis.sh stop

运行redis：

./redis.sh start

重新配置redis:

./redis.sh reload

连接Redis Cluster：

redis-cli -c -h 172.17.169.196 -p 7000 // -c参数为集群操作

### 1.7、如果是在阿里云或者是公网上面搭建redis需要进行一下操作

1. 注意安装集群的时候，防火墙中不仅要放开redis的端口，还要放开，而且需要开通集群总线端口

集群总线端口为redis客户端连接的端口 +10000

如redis端口为7000

则集群总线端口为17000

所以当前网络要开放17000到17005端口



2.当java程序连接redis集群的时候，jedis-2.9.0.jar这个源包里面要取得是公网的内网ip，所以会出错，要用修改过源码的包jedis-2.9.0.jar 覆盖下maven本地仓库下的jar包。

Maven pom.xml里面去除依赖方式：

<dependency>    
     <groupId>org.springframework.boot</groupId>    
     <artifactId>spring-boot-starter-data-redis</artifactId>   
  <exclusions>      
                  <exclusion>      
         <groupId>redis.clients</groupId>      
         <artifactId>jedis</artifactId>      
         </exclusion>    
       </exclusions>   
 </dependency>   
   
 <dependency>  
   <groupId>redis.clients</groupId>  
   <artifactId>jedis</artifactId>  
   <version>2.7.2</version>

</dependency>

查看阿里云依赖包pom文件的写法地址：http://maven.aliyun.com/nexus/#nexus-search;quick~redis.clients

进行完以下方式后应该就能成功连接了。

Adding a new node

```

redis-cli --cluster add-node 127.0.0.1:7006 127.0.0.1:7000

```

Adding a new node as a replica

```

redis-cli --cluster add-node 127.0.0.1:7006 127.0.0.1:7000 --cluster-slave

```

添加从节点到指定主节点

```

redis-cli --cluster add-node 127.0.0.1:7006 127.0.0.1:7000 --cluster-slave --cluster-master-id 3c3a0c74aae0b56170ccb03a76b60cfe7dc1912e

```

查看主从关系

```

cluster nodes

info replication

redis-cli --cluster check IP:Port

```

迁移槽和数据

```

redis-cli --cluster reshard 192.168.2.107:7001

```

Removing a node

```

redis-cli --cluster del-node 127.0.0.1:7000 `<node-id>`

```

The first argument is just a random node in the cluster, the second argument is the ID of the node you want to remove.

Replicas migration

```

CLUSTER REPLICATE <master-node-id>

```

Upgrading nodes in a Redis Cluster

```

```

### 平衡各个节点槽的数量

```

# ./src/redis-cli --cluster rebalance --cluster-threshold 1 192.168.25.64:6380

```