DTCC

2017第八届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2017

大规模redis集群的服务治理之路

高嵩

About me

- 11年工作经验，2011年加入优酷，曾任高级工程师、技术专家，现任高级技术经理
- 目前在大优酷数据战略团队主要负责分布式缓存、实时计算平台的搭建与优化
- 对分布式存储、流计算、高并发高可用系统有浓厚兴趣，热爱分享与交流
- 技术博客：http://blueswind8306.iteye.com/

目录

- RedisCluster介绍
- RedisCluster优（cǎi）化（kēng）经验
- RedisCluster运维经验
- RedisCluster服务化

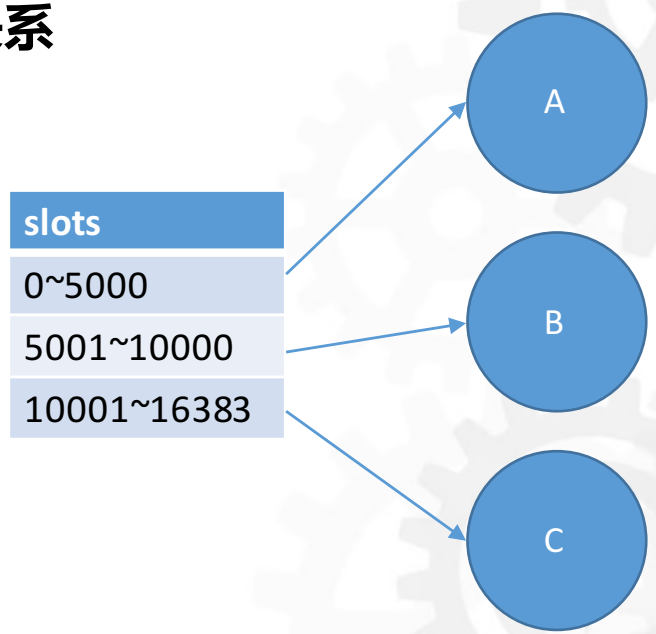
RedisCluster介绍

Redis Cluster特性

- 支持string/hash/list/set/sortedset/hyperloglog/geo/pubsub等大部分单机Redis功能
- 去中心化的分布式集群
- 主从全量/增量同步
- 服务端分片
- 节点水平伸缩，扩容/缩容对调用方透明
- 自动failover/failback

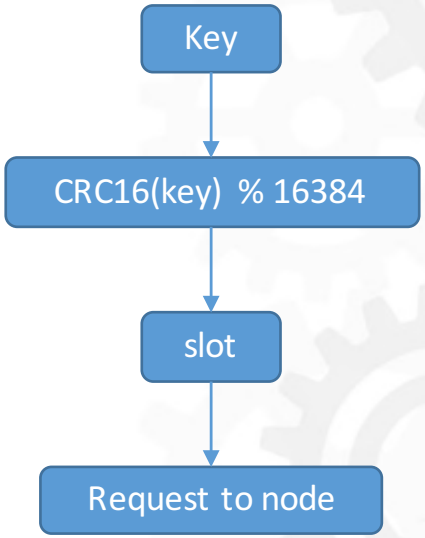
服务端分片

- 数据分为固定的16384个槽（slot）
- 每个node负责一部分slot的数据存储
- 每个node有整个集群的slot->node映射关系
- 集群初始化时确定slot->node的关系
- 扩容/缩容通过slot迁移完成



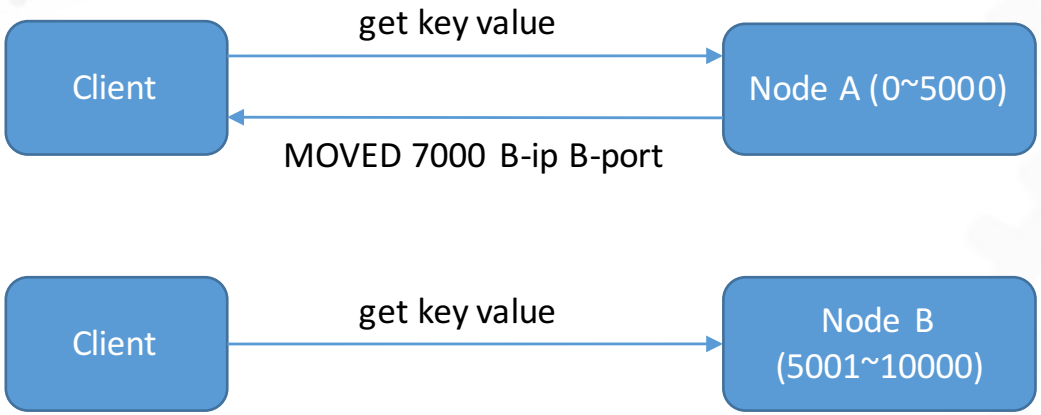
客户端请求

- 客户端缓存slot->node的映射关系
- 请求一个key时，在本地先算出key对应的slot，再根据slot->node的对应关系找到node
- 如果服务端的映射关系和客户端不一致怎么办？



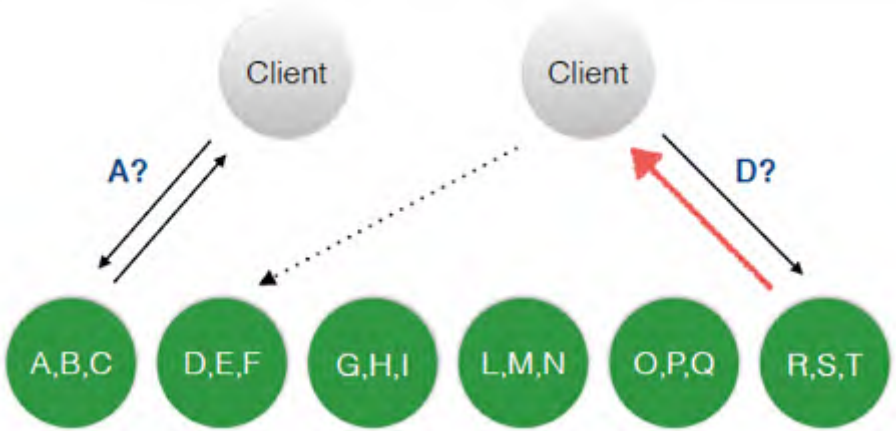
请求重定向 – MOVED

- 客户端请求的key所对应的槽不在Node A
- Node A会返回一个长期重定向错误（MOVED）
- 客户端根据MOVED重定向信息，访问Node B
- 客户端重新缓存slot->node的映射关系



请求重定向

- 解决了客户端与服务端的一致性问题
- 集群状态变化（扩容/缩容）对客户端请求不会造成影响
- 支持多次跳转



Redis Cluster在优酷

- **峰值QPS：800w+**
- **实例数：500+**
- **内存：2T/4T**

RedisCluster优（cǎi）化（kēng）经验

网卡绑定

- **网卡中断造成的影响**
  - ✓ 抢占Redis进程CPU、降低Redis吞吐
- **解决方法**
  - ✓ 关闭irqbalance
  - ✓ 将网卡中断绑定到固定的CPU
  - ✓ 将服务进程绑定到其它核

网卡绑定

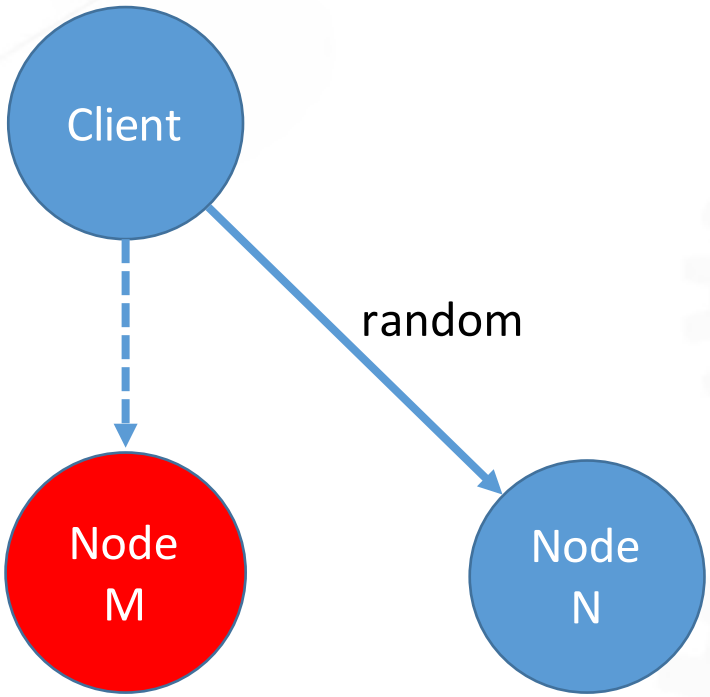
- **绑定后效果**

```
top - 22:01:17 up 10 days, 10:12,  1 user,  load average: 0.02, 0.02, 0.05
Tasks: 225 total,   9 running, 216 sleeping,   0 stopped,   0 zombie
Cpu0  :  1.6%us,  0.5%sy,  0.0%ni, 74.0%id,  0.0%wa,  0.5%hi, 23.4%si,  0.0%st
Cpu1  :  1.2%us,  1.2%sy,  0.0%ni, 32.3%id,  0.0%wa,  0.6%hi, 64.6%si,  0.0%st
Cpu2  :  1.7%us,  1.7%sy,  0.0%ni, 33.9%id,  0.0%wa,  0.0%hi, 62.8%si,  0.0%st
Cpu3  : 45.3%us, 43.6%sy,  0.0%ni, 10.7%id,  0.0%wa,  0.0%hi,  0.3%si,  0.0%st
Cpu4  : 46.5%us, 41.8%sy,  0.0%ni, 11.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu5  : 33.8%us, 35.1%sy,  0.0%ni, 31.1%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu6  : 45.3%us, 37.0%sy,  0.0%ni, 17.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu7  : 47.0%us, 39.5%sy,  0.0%ni, 13.5%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu8  : 46.3%us, 41.2%sy,  0.0%ni, 12.5%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu9  : 48.3%us, 41.7%sy,  0.0%ni, 10.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu10 : 47.0%us, 40.5%sy,  0.0%ni, 12.5%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu11 :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  132174492k total,  2912544k used, 129261948k free,   180264k buffers
Swap:  4194300k total,        0k used,  4194300k free,   939360k cached
```

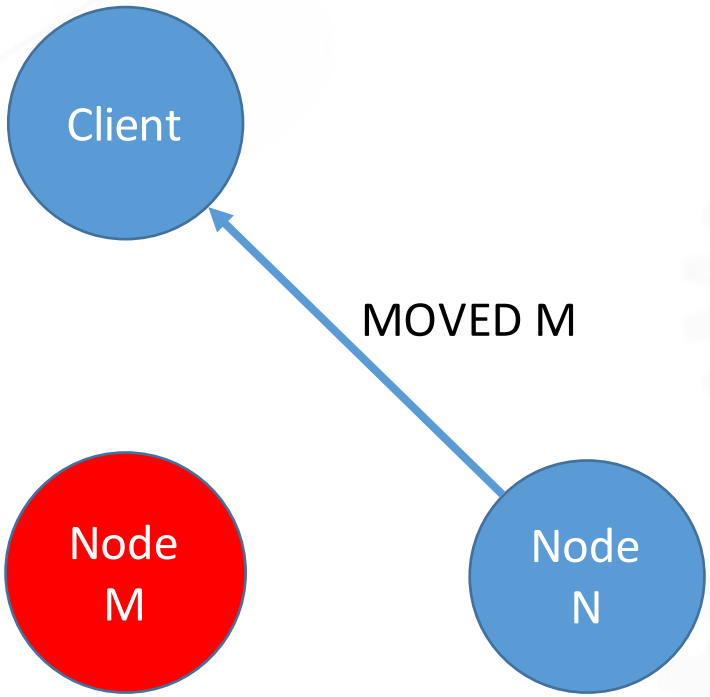
Slot风暴

- **RT耗时变长**
- **通过监控发现cluster slots命令被频繁调用**

Slot风暴

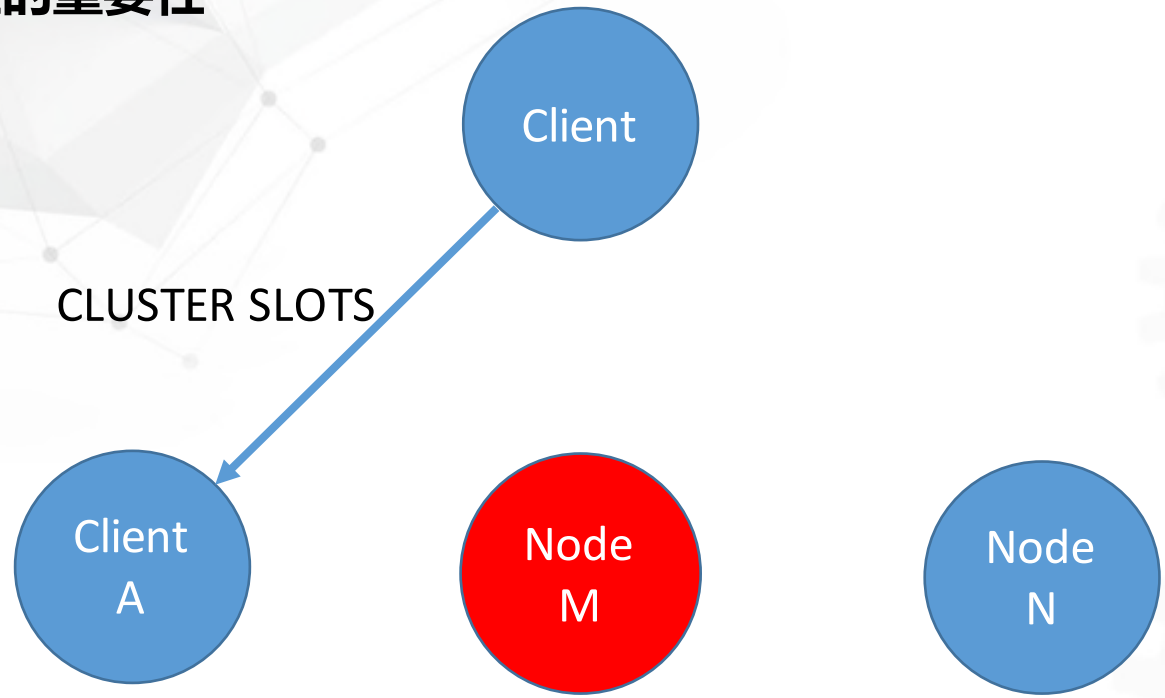


Slot风暴



Slot风暴

- **Jedis-2.8.0以上版本修复了该bug，增加shuffle逻辑**
- **监控的重要性**



其它优化参数

- **内存相关**
  - ✓ maxmemory-policy volatile-ttl
  - ✓ hz 100（当过期key较多时）
- **主从同步**
  - ✓ repl-backlog-size
  - ✓ client-output-buffer-limit slave
- **集群相关**
  - ✓ cluster-require-full-coverage no
  - ✓ cluster-node-timeout

Redis Cluster运维经验

集群扩容

- **将新实例加入集群**
  - ✓ redis-trib.rb add-node 新masterIP:port 现有节点IP:port
  - ✓ redis-trib.rb add-node --slave --master-id <master-id> 新slaveIP:port 现有节点IP:port
- **扩容目标**
  - ✓ 在key分布大致均衡的前提下，尽量使每个实例的slot个数一致

集群扩容

Node A	Node B	Node C
0	4	
1	5	
2	6	
3	7	
	8	

集群扩容

- 按照slot个数排序
- 算出平均slot数

Node B	Node A	Node C
4	0	
5	1	
6	2	
7	3	
8		

集群扩容

- **按照slot个数排序**
- **算出平均slot数**
- **将大于平均值的节点的槽迁移到新节点**

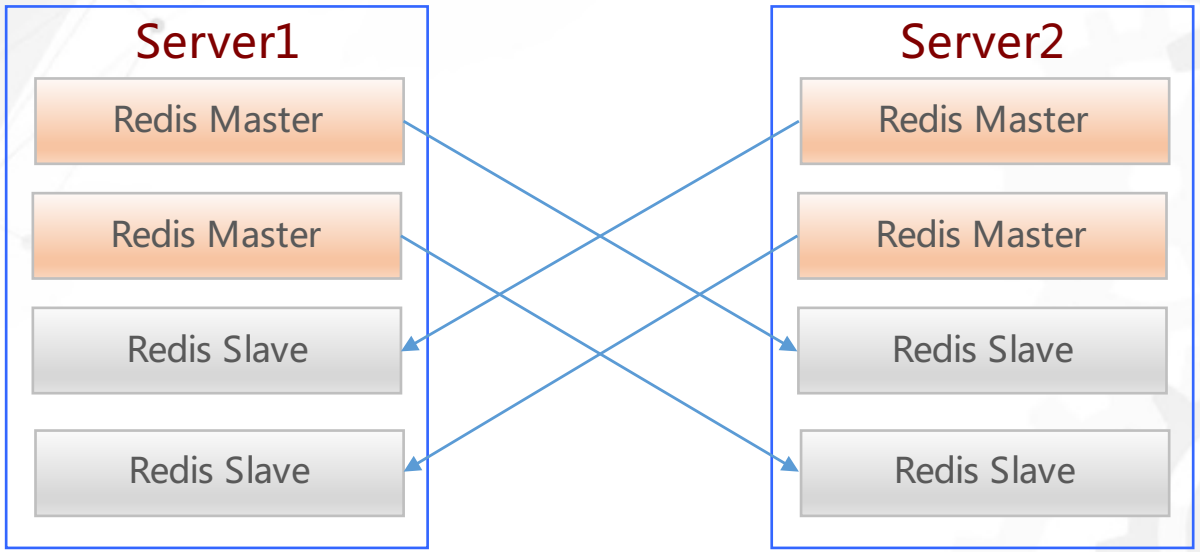
Node B	Node A	Node C
6	1	4
7	2	5
8	3	0

  - ✓ redis-trib.rb reshard --from <node-id> --to <node-id> --slots <numslots> --yes <host>:<port>

- **Redis 3.2开始，redis-trib.rb支持rebalance功能了！**
  - ✓ 支持权重、阈值配置
  - ✓ 对于新扩容节点需要增加 --use-empty-masters参数
  - ✓ demo
  - ✓ 说明文档

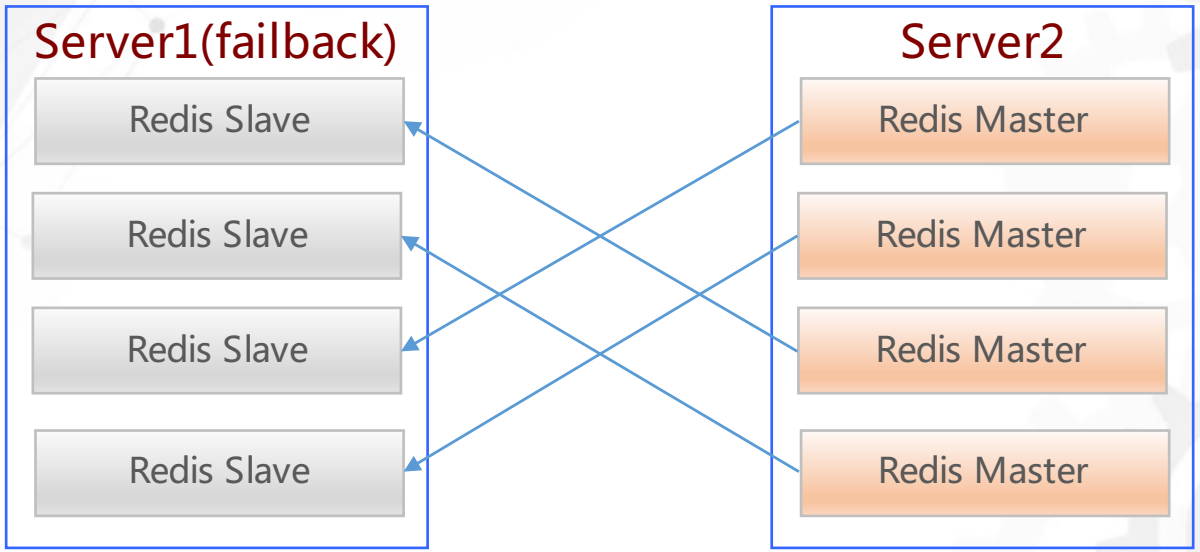
手动failover

- **命令：cluster failover**
- **场景1：节点故障恢复后，需要恢复之前的主从关系**



手动failover

- **命令：cluster failover**
- **场景1：节点故障恢复后，需要恢复之前的主从关系**



手动failover

- **场景2：集群子版本升级**
- **步骤1：关闭从节点->升级->重启从节点**
  - ✓ 注意：重启后需要等待主从重新同步完成
- **步骤2：对新版本实例做手动failover->新主晋升完成**
- **重复以上两步，直到集群所有实例升级完成**
- **注意：此操作可能会有新老版本兼容性风险，必须充分测试！**

Redis Cluster服务化

服务化调用

- **Proxy层**
  - ✓ 跨语言支持
  - ✓ Redis层连接数可控
  - ✓ 安全性（鉴权、屏蔽危险命令）
  - ✓ 批量接口支持
  - ✓ 多租户资源隔离
  - ✓ 配额限制

服务治理

- **热key**
  - ✓ 客户端Localcache（一致性？）
  - ✓ Key打散
- **大key**
  - ✓ Limit限制（key大小、zset自动淘汰）
  - ✓ 动态压缩
- **死key**
  - ✓ 统一设置key过期

监控告警

- **应用级监控**
  - ✓ 接口QPS / RT / 错误率 / 命中率
  - ✓ 多租户下的内存分析
- **Redis监控（Open-falcon + redismon）**
  - ✓ 客户端连接：blocked_clients / connected_clients / connected_clients_pct
  - ✓ 集群状态：cluster_slots_pfail / cluster_slots_fail
  - ✓ 命令调用： total_commands_processed / cmdstat_xxx / slowlog_len
  - ✓ 内存使用：used_memory / used_memory_pct / mem_fragmentation_ratio
  - ✓ 主从同步：master_link_status / sync_full

总结

- **RedisCluster介绍**
  - ✓ RedisCluster特性
  - ✓ 服务端分片
  - ✓ 客户端请求
  - ✓ 请求重定向
- **RedisCluster优（cǎi）化（kēng）经验**
  - ✓ 网卡绑定
  - ✓ slot风暴
  - ✓ 其它优化参数
- **RedisCluster运维经验**
  - ✓ 集群扩容
  - ✓ 手动failover
- **RedisCluster服务化**
  - ✓ 服务化调用
  - ✓ 服务治理
  - ✓ 监控告警


THANKS
SequeMedia
盛拓传媒
IT168.com
ITPUB
ChinaUnix