

欢迎加入向上IT技术交流群

始于2011年

工作，学习，友谊
草根IT技术交流，1月1次，面对面
<http://www.eeqee.com>



NoSQL总群:
224718662

技术支持联系:
QQ 42950223

微信—>发现—>扫一扫

redis探索之旅

成锁元

2013-10-15

redis探索之旅

- ▶redis概述
- ▶redis 缓存 or DB
- ▶redis与memcache比较
- ▶redis基础数据结构
- ▶redis持久化机制
- ▶redis容量规划
- ▶reids线上问题分析

redis概述

- 快速、轻量级键值对数据存储

• redis宣言：

- 1\操作数据结构语言的数据工具
- 2\定位于内存数据库
- 3\API简洁易用
- 4\诗一般优美的代码
- 5\始终避免复杂化
- 6\支持两个层次的API
- 7\以优化代码为乐

redis概述

- redis不足:

- 不是Big Data
- 集群的解决方案不成熟
- 不支持灵活的sql查询

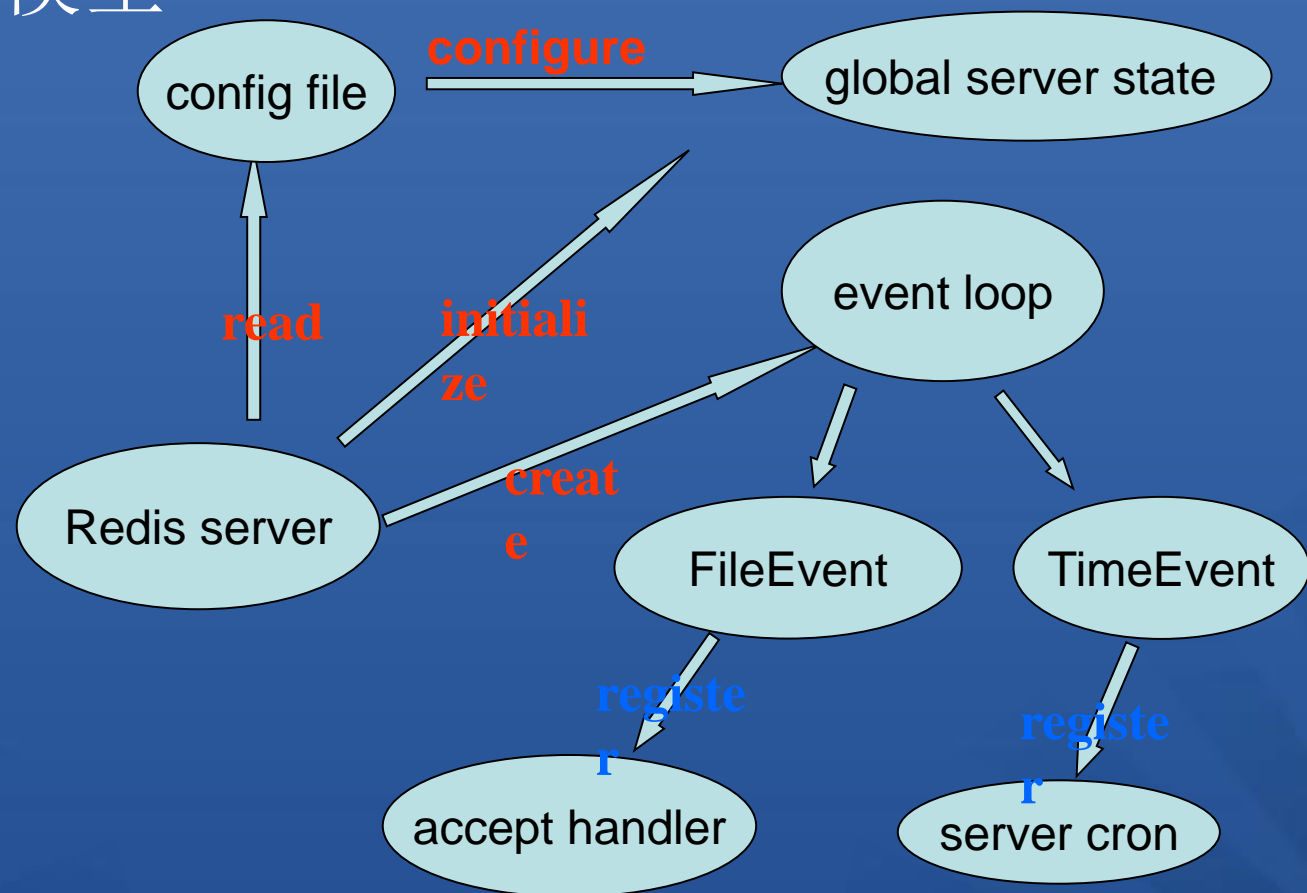
缓存 OR DB ?

redis与memcache比较

	redis	memcache	
数据类型	多种结构	字符串	
事件驱动模型	AE事件驱动	libEvent	
支持存储	支持	不支持	
其他	支持复制 单进程单线程 Crash safe Recovery slow	多线程 不能枚举全数据 性能仍有提升空间	

redis 基础数据结构

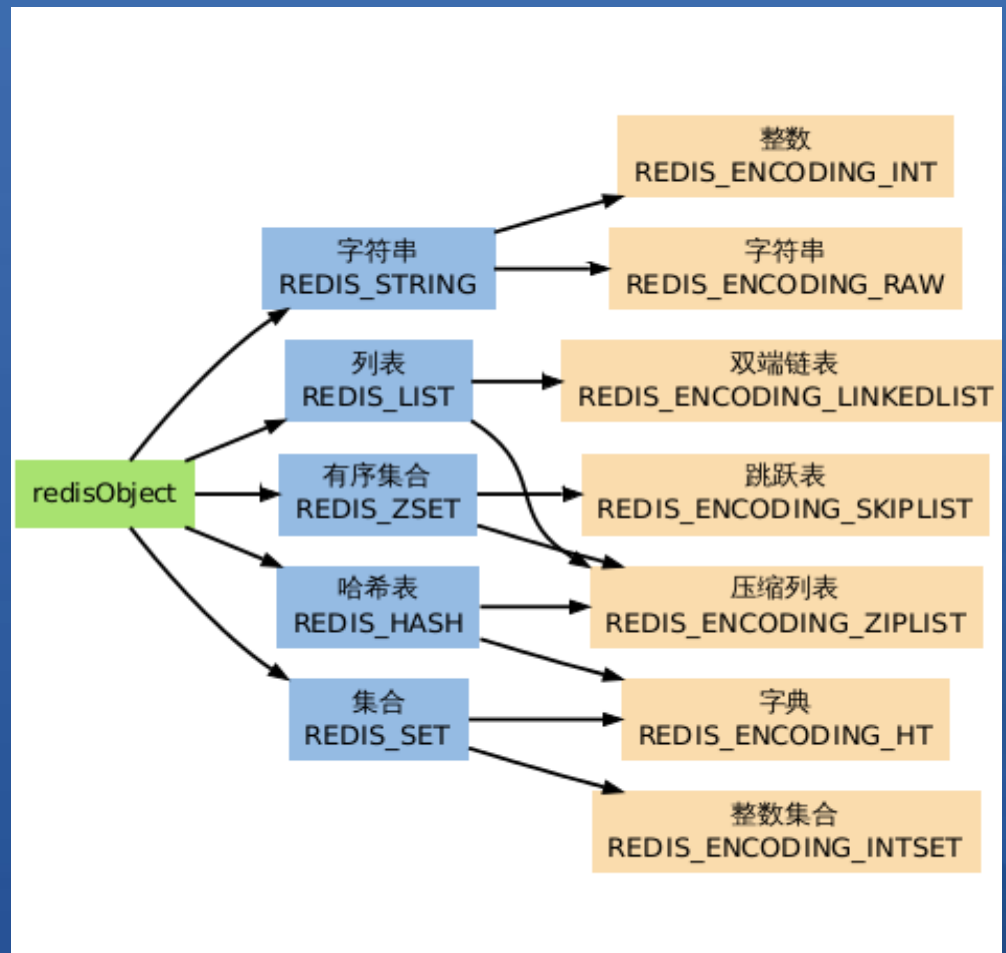
• AE 事件模型



redis 基础数据结构

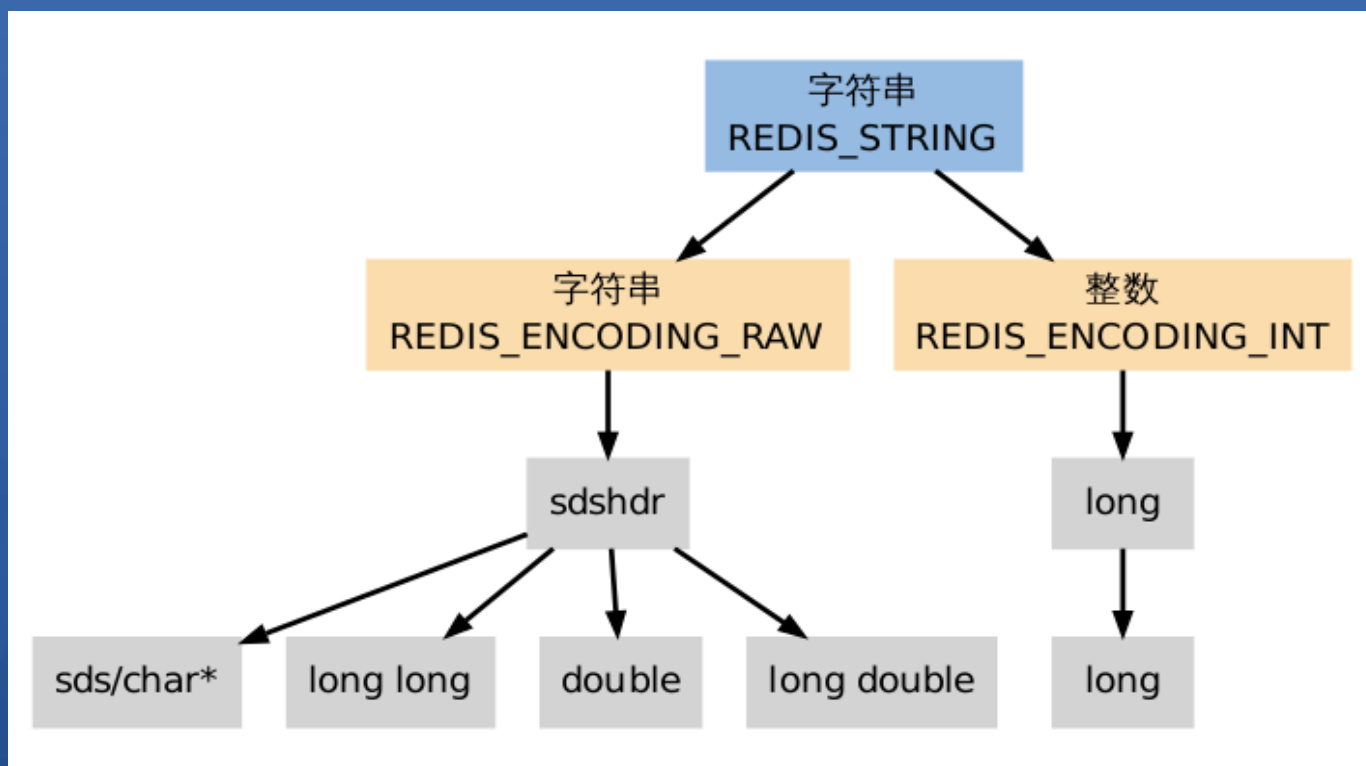
- 数据类型

- 字符串 (**String**)
- 哈希表 (**zipmap**)
- 列表 (**ziplist**)
- 集合 (**set**)
- 有序集 (**sortedset**)



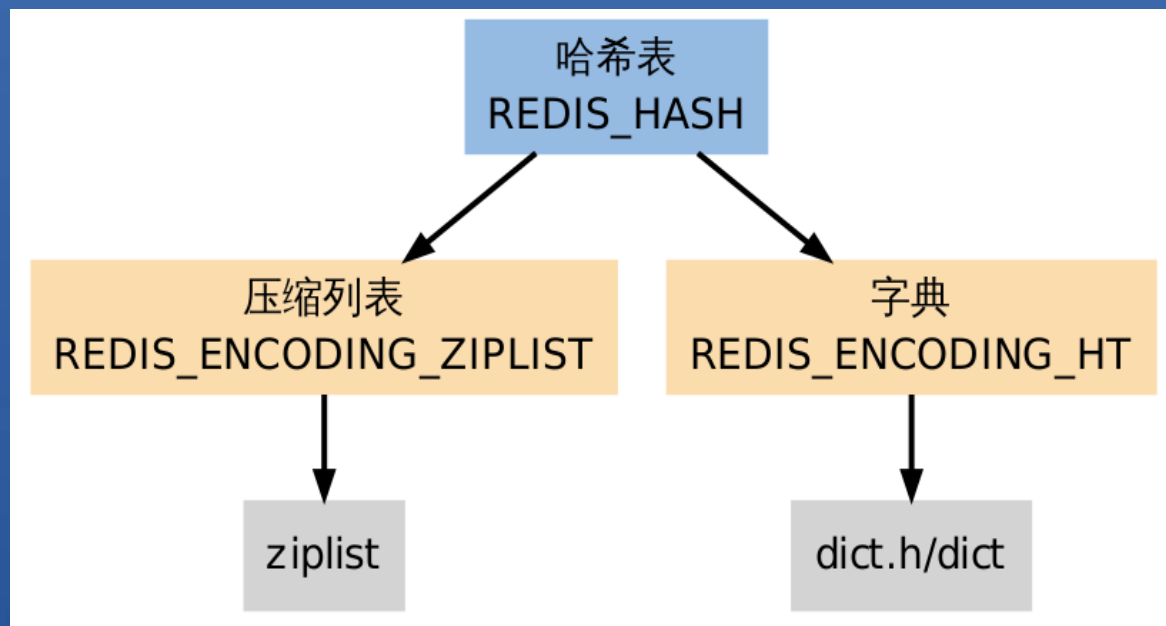
redis基础数据结构

- 数据类型-字符串



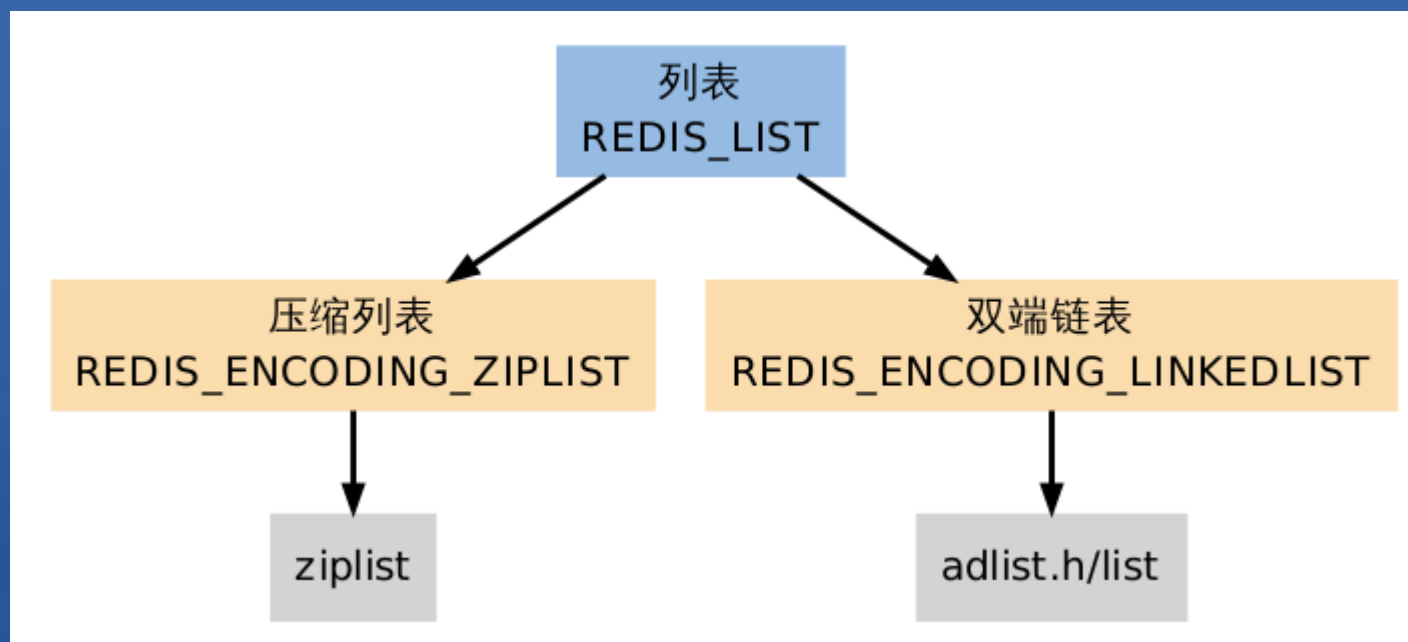
redis基础数据结构

- 数据类型-哈希表



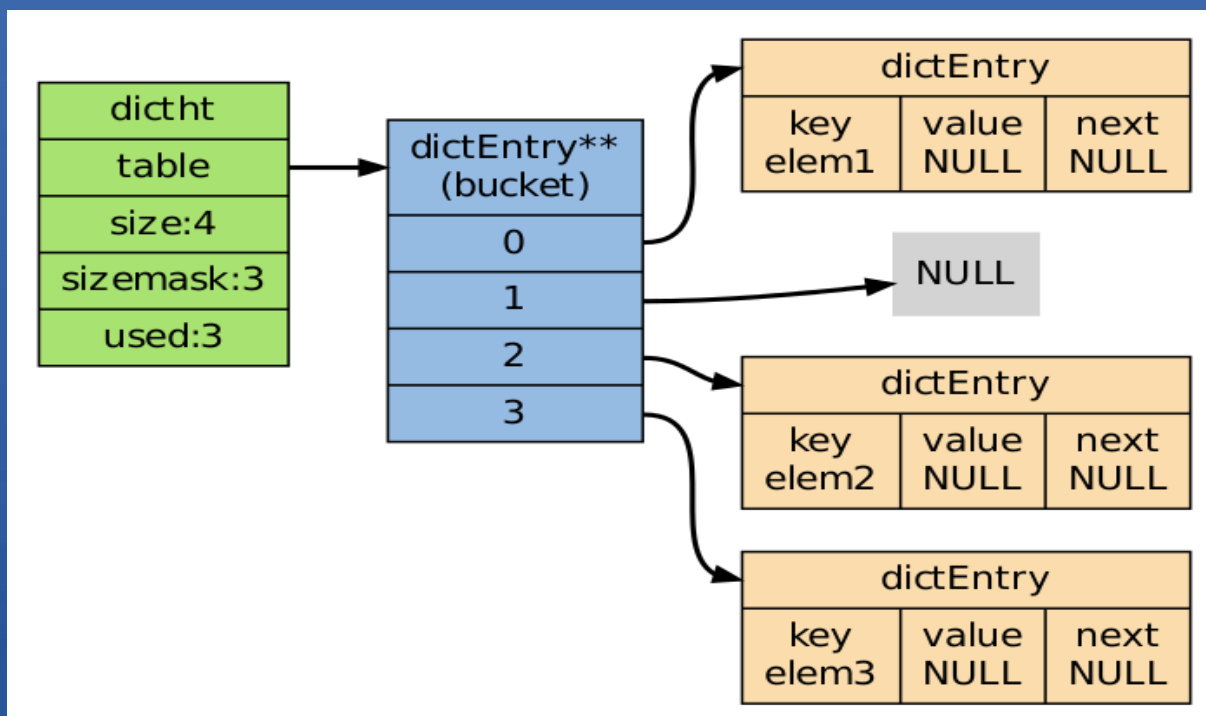
redis基础数据结构

- 数据类型-列表



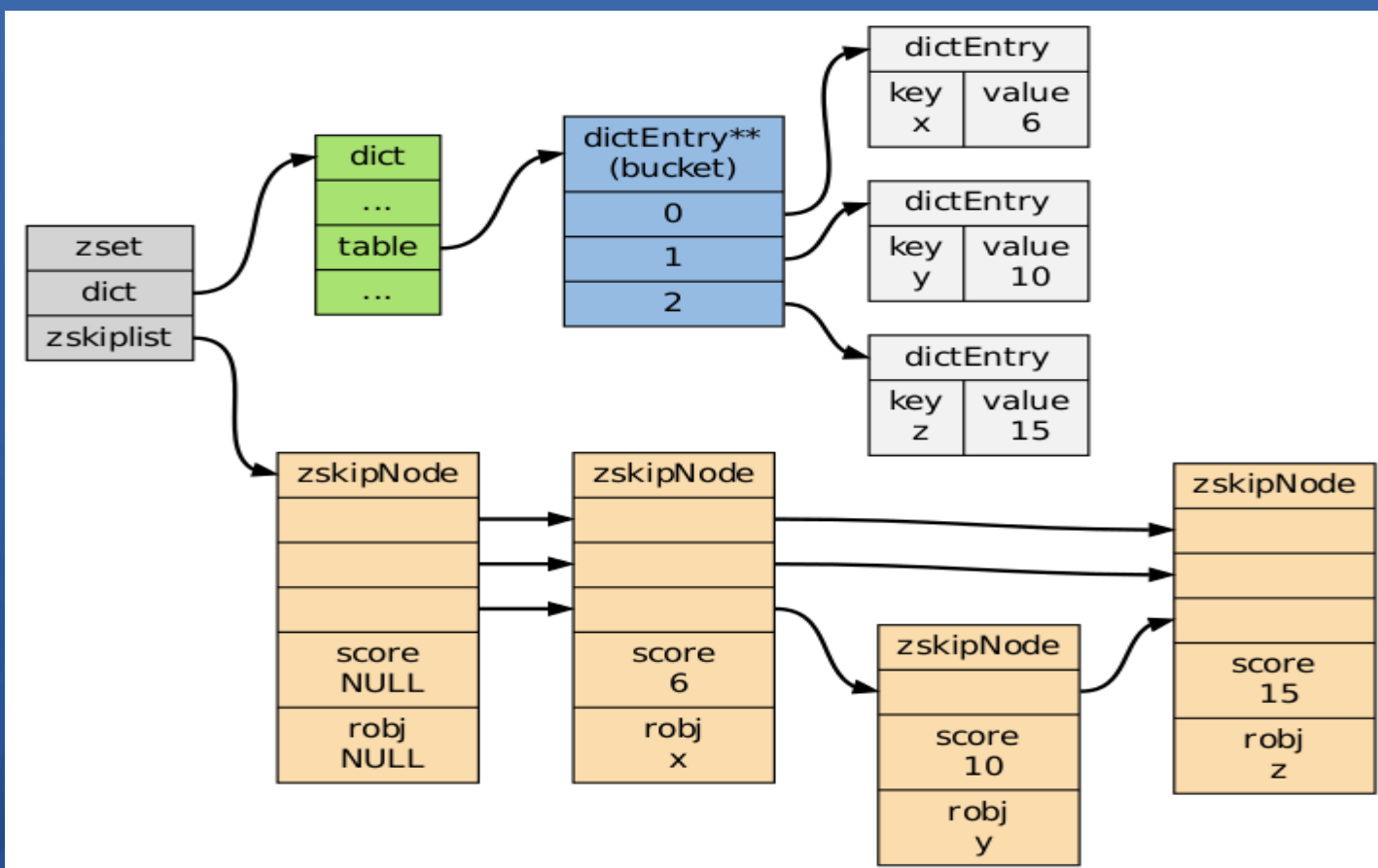
redis基础数据结构

- 数据类型-集合



redis基础数据结构

- 数据类型-有序集



Redis 事务

- 指令

Multi

Exec

Discard

- 不足

不能保证原子性

redis持久化机制

- Snapshot

save 900 1

save 300 10

save 60 10000

- AOF

appendfsync always

appendfsync everysec //在性能和持久化方面做了很好的折中

appendfsync no

redis持久化机制

- 建议

- 更新频繁，一致性要求比较高，**AOF**策略为主
- 更新不频繁，可以容忍少量数据丢失或错误，**Snapshot**策略为主

redis容量规划

- 字符串类型计算

- 计算公式:

- 键值个数 * (dictEntry大小 + redisObject大小 + 包含key的sds大小 + 包含value的sds大小) + bucket字节数
 - 说明:
 - 32位系统: dicEntry 16字节, reidsObject 16字节, key 或 value +9 最接近的2的N次方个字节, bucket数*4 字节
 - 64位系统: dicEntry 24字节, reidsObject 24字节, key 或 value +9 最接近的2的N次方个字节, bucket数*8字节

redis容量规划

• Hash类型计算

– 计算公式:

- 键值个数 * (dictEntry大小 + redisObject大小 + 包含key的sds大小 + subkey的总大小) + bucket字节数
- 说明:
 - 32位系统: dicEntry 16字节, reidsObject 16字节, key +9 小于等于2的N次方个字节, bucket数*4 字节
 - 64位系统: dicEntry 24字节, reidsObject 24字节, key +9 小于等于2的N次方个字节, bucket数*8字节
 - subkey的总大小: sub个数 * (subkey+subval) +2 (zm头)

redis线上问题分析

- 应用场景

- 计数服务

- 需求: 1000万用户的计数 (5种计数)

BPtest(redis1)
master
Address: 10.70.33.58:7007
Slave:
memory usage: 443.77M
number of connected slaves: 0
uptime in days: 2
uptime in seconds: 256514
connected clients: 2
toggle detail

BPtest(redis2)
master
Address: 10.70.33.59:7007
Slave:
memory usage: 471.27M
number of connected slaves: 0
uptime in days: 8
uptime in seconds: 709502
connected clients: 2
toggle detail

BPtest(redis3)
master
Address: 10.70.33.60:7007
Slave:
memory usage: 456.03M
number of connected slaves: 0
uptime in days: 8
uptime in seconds: 708001
connected clients: 2
toggle detail

BPtest(redis4)
master
Address: 10.70.33.68:7007
Slave:
memory usage: 435.50M
number of connected slaves: 0
uptime in days: 8
uptime in seconds: 707824
connected clients: 2
toggle detail

redis线上问题分析

STATUS
master

USED MEMORY
419.67M

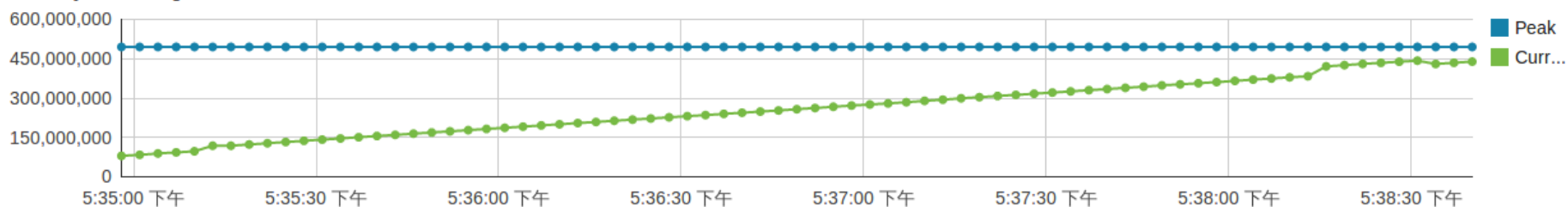
TOTAL KEYS
2.3M

CLIENTS
3

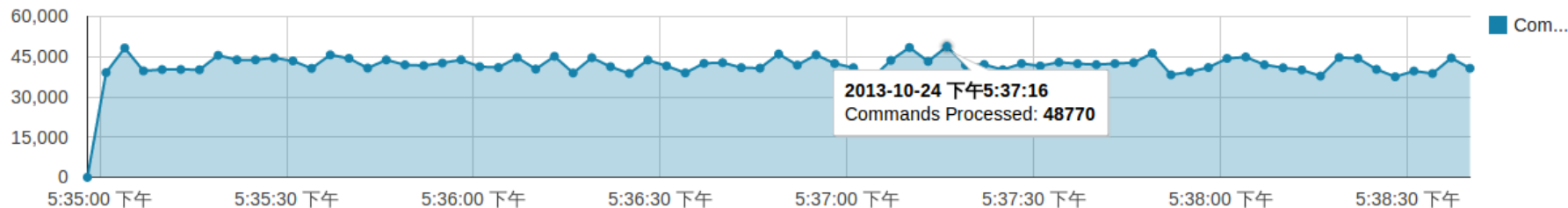
COMMANDS PROCESSED
24.2M

UPTIME
8.3d

Memory Consumption



Commands Processed



redis线上问题分析

- sharding

- 通过客户端sharding, 如jedis

- twemproxy

缺点都解决不了自动re-sharding

redis线上问题分析

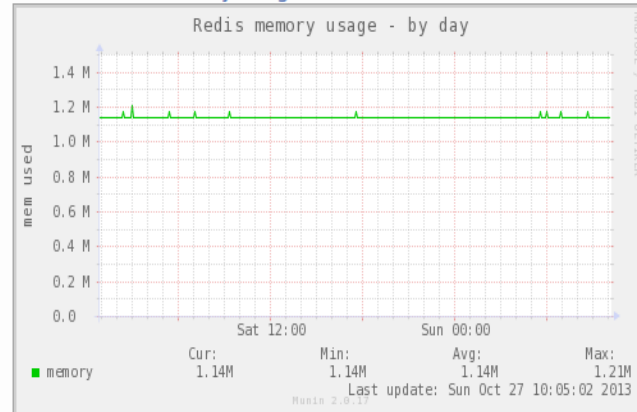
- 监控
 - Redis-monitor
 - munin, nagios等

redis线上问题分析

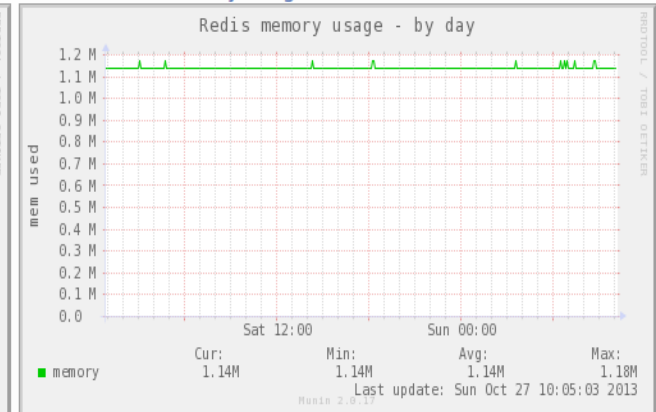
• munin

Redis memory usage

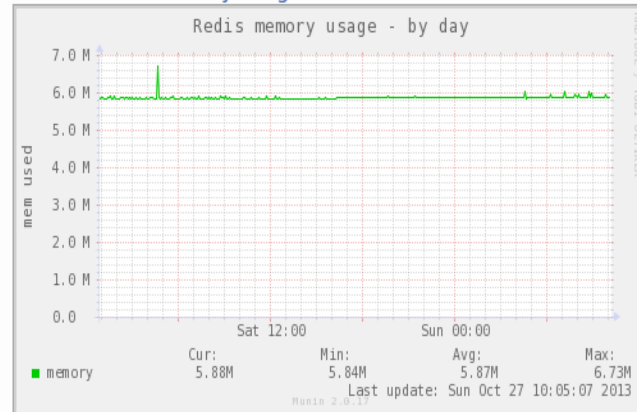
mc1 :: Redis memory usage



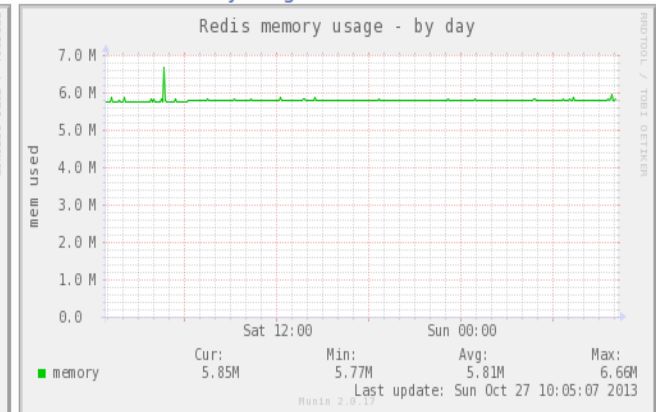
mc2 :: Redis memory usage



mc3 :: Redis memory usage

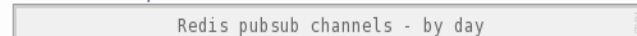


mc4 :: Redis memory usage

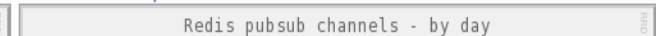


Redis pubsub channels

mc1 :: Redis pubsub channels



mc2 :: Redis pubsub channels



redis线上问题分析

- 教训
 - 不要启动到虚拟机上
 - 慎用 **keys mXXX**, **save**, **bgsave**, **bgrewriteaof**, **monitor**
 - 安全：禁用客户端特殊命令：**flushall debug flushDB**
 - 不建议动态配置

参考资料

- 作者Antirez的博客 <http://antirez.com/>
- redis设计与实现
<http://www.redisbook.com/>
- 源码与注释
[https://github.com/huangz1990/annotated_redis_source/](https://github.com/huangz1990/annotated_redis_source)

欢迎加入向上IT技术交流群

始于2011年

工作，学习，友谊
草根IT技术交流，1月1次，面对面
<http://www.eeqee.com>



NoSQL总群:
224718662

技术支持联系:
QQ 42950223

微信—>发现—>扫一扫