分布式缓存服务

性能白皮书

文档版本 01

发布日期 2024-04-07

华为云计算技术有限公司

版权所有 © 华为云计算技术有限公司 2024。 保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明

HUAWEI和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 测试方法.....1
1.1 redis-cli 和 redis-benchmark 测试工具介绍.....1
1.2 使用 redis-benchmark 工具进行性能测试.....3
1.3 使用 memtier_benchmark 工具进行性能测试.....5
1.4 redis-benchmark 与 memtier_benchmark 的差异.....7
2 Redis 3.0 主备实例测试数据.....8
3 Redis 3.0 Proxy 集群实例测试数据.....10
4 Redis 4.0/5.0 主备实例测试数据.....12
5 Redis 4.0/5.0 Proxy 实例测试数据.....15
6 Redis 4.0/5.0 Cluster 集群实例测试数据.....17
7 Redis 6.0 主备实例测试数据.....19
8 Redis 6.0 Cluster 集群实例测试数据.....22
9 Redis 备份恢复迁移性能测试数据.....25

1 测试方法

1.1 redis-cli 和 redis-benchmark 测试工具介绍

获取 redis-cli 和 redis-benchmark

创建弹性云服务器（ECS），根据不同的操作系统直接安装Redis-server，下面以ubuntu和CentOS系统为例：

说明

直接编译安装Redis或者使用yum，apt安装Redis-server，安装Redis-Server的同时，会同步安装benchmark。

- ubuntu系统
```
sudo apt update
sudo apt install redis-server
```
- CentOS系统
```
sudo yum install epel-release
sudo yum update
sudo yum -y install redis
```

也可以直接下载安装包，进行解压和编译，以下载redis-6.0.9版本为例：

1. 下载redis-6.0.9客户端。

   wget http://download.redis.io/releases/redis-6.0.9.tar.gz
2. 解压客户端压缩包。

   tar xzf redis-6.0.9.tar.gz
3. 进入redis-6.0.9的src目录下。

   cd redis-6.0.9/src
4. 编译源码。

   make

   编译完成后，工具一般在redis-x.x.x的src目录下。

redis-cli 常用选项

- -h <hostname>： 服务器的主机名，可以是IP或者域名

版权所有 © 华为云计算技术有限公司

- -p <port>： 服务器的端口，默认是6379
- -a <password>： 连接服务器的密码，免密连接的实例无需输入-a {password}
- -r <repeat>： 执行指定命令N次
- -n <db>： 数据库编号，默认是0
- -c： 启用集群模式（遵循-ASK和-MOVED重定向）
- --latency： 进入特殊模式连续采样延迟
- --scan： 非阻塞式地扫描键空间（区别于keys *扫描键空间会导致redis-server阻塞）
- --eval <file>： 使用Lua脚本发送EVAL命令
- -x： 读取STDIN中的最后一个参数
- --bigscan： 扫描数据集的大key
- --raw： 如果显示数据是这样的'\xe4\xb8'十六进制编码，可以尝试加--raw转为直接显示原始数据

了解redis-cli更多详情，请访问 **https://redis.io/docs/connect/cli/**

redis-cli 常用命令举例

- 连接实例：
  ./redis-cli -h {IP} -p 6379
- 指定连接某个DB：
  ./redis-cli -h {IP} -p 6379 -n 10
- 连接cluster集群实例：
  ./redis-cli -h {IP} -p 6379 -c
- 测试时延（原理是发ping命令）：
  ./redis-cli -h {IP} -p 6379 --latency
- 执行scan扫描匹配指定模式的key：
  ./redis-cli -h {IP} -p 6379 --scan --pattern '*:12345*'

redis-benchmark (redis-6.0.9)常用选项

- -h <hostname>： 服务器的主机名，可以是IP或者域名
- -p <port>： 服务器的端口，默认是6379
- -a <password>： 连接服务器的密码，免密连接的实例无需输入-a {password}
- -c <clients>： 并发连接数 ，默认50
- -n <requests>： 请求总数（默认为100000）
- -d <size>： SET/GET值的数据大小（以字节为单位，默认值2）
- --dbnum <db>： 选择指定的数据库编号（默认值0）
- --threads <num>： 启动多线程模式（redis 6.0版本编译的redis-benchmark才支持，多线程压测redis的性能优于单线程）
- --cluster： 启动集群模式（cluster集群才需要该参数）
- -k <boolean>： 1=keep alive 0=reconnect（默认值1，可以测试长短连接）
- -r <keyspacelen>： 对SET/GET/INCR使用随机键，对SADD使用随机值。参数中keyspacelen 指的是添加键的数量。

- -e： 如果服务器回复错误，请在stdout上显示它们
- -q： 只展示query/sec的值
- -l： 循环测试
- -t <tests>： 可以对指定的命令进行测试
- -I： 空闲模式。仅打开N个空闲连接并等待
- -P <numreq>： 管道请求的并发数量（默认值为1）

了解redis-benchmark更多详情，请访问：**https://redis.io/docs/reference/optimization/benchmarks/**

redis-benchmark 常用命令举例

- 单机、主备、读写分离和proxy集群的测试命令：

  ./redis-benchmark -h {IP或域名} -p 6379 -a {password} --threads {num} -n { nreqs } -r { randomkeys } -c {clients} -d {datasize} -t {command}
- cluster集群测试命令：

  ./redis-benchmark -h {IP或域名} -p 6379 -a {password} --threads {num} -n { nreqs } -r { randomkeys } -c {clients} -d {datasize} --cluster -t {command}
- 测试短连接：

  ./redis-benchmark -h {IP或域名} -p 6379 -a {password} --threads {num} -n { nreqs } -r { randomkeys } -c {clients} -d {datasize} -k 0 -t {command}
- 测试空闲连接：

  ./redis-benchmark -h {IP或域名} -p 6379 -a {pwd} -c {clients} -I

1.2 使用 redis-benchmark 工具进行性能测试

针对DCS Redis实例的性能评估，比如测试某种实例规格的set或get在某个高并发场景下的性能，可参考本文的测试方法。

测试工具

Redis客户端源码包含一个名为redis-benchmark的性能测试工具，它可以模拟N个客户端同时向Redis发送M条查询命令的应用场景。

redis-benchmark工具的使用方法请参考**redis-cli和redis-benchmark测试工具介绍**。

下载与安装工具

1. 下载redis-6.0.9客户端。

   wget http://download.redis.io/releases/redis-6.0.9.tar.gz
2. 解压客户端压缩包。

   tar xzf redis-6.0.9.tar.gz
3. 进入redis-6.0.9的src目录下。

   cd redis-6.0.9/src
4. 编译源码。

   make

5. 查看是否有redis-benchmark可执行文件

   ls

```
[root@[redacted] src]# ls
adlist.c      config.h          geohash_helper.h  lzfP.h          rax.o             scripting.o     t_hash.c
adlist.h      config.o          geohash_helper.o  Makefile        rdb.c             sdsalloc.h      t_hash.o
adlist.o      crc16.c           geohash.o         memtest.c       rdb.h             sds.c           t_list.c
ae.c          crc16.o           geo.o             memtest.o       rdb.o             sds.h           t_list.o
ae_epoll.c    crc64.c           help.h            mkreleasehdr.sh redisassert.h     sds.o           t_set.c
ae_evport.c   crc64.h           hyperloglog.c     module.c        redis-benchmark   sentinel.c      t_set.o
ae.h          crc64.o           hyperloglog.o     module.o        redis-benchmark.c sentinel.o      t_stream.c
ae_kqueue.c   db.c              intset.c          modules         redis-benchmark.o server.c        t_stream.o
ae.o          db.o              intset.h          multi.c         redis-check-aof   server.h        t_string.c
ae_select.c   debug.c           intset.o          multi.o         redis-check-aof.c server.o        t_string.o
anet.c        debugmacro.h      latency.c         networking.c    redis-check-aof.o setproctitle.c  t_zset.c
anet.h        debug.o           latency.h         networking.o    redis-check-rdb   setproctitle.o  t_zset.o
anet.o        defrag.c          latency.o         notify.c        redis-check-rdb.c sha1.c          util.c
aof.c         defrag.o          lazyfree.c        notify.o        redis-check-rdb.o sha1.h          util.h
aof.o         dict.c            lazyfree.o        object.c        redis-cli         sha1.o          util.o
asciilogo.h   dict.h            listpack.c        object.o        redis-cli.c       siphash.c       valgrind.sup
atomicvar.h   dict.o            listpack.h        pqsort.c        redis-cli.o       siphash.o       version.h
bio.c         endianconv.c      listpack_malloc.h pqsort.h        redismodule.h     slowlog.c       ziplist.c
bio.h         endianconv.h      listpack.o        pqsort.o        redis-sentinel    slowlog.h       ziplist.h
bio.o         endianconv.o      localtime.c       pubsub.c        redis-server      slowlog.o       ziplist.o
bitops.c      evict.c           localtime.o       pubsub.o        redis-trib.rb     solarisfixes.h  zipmap.c
bitops.o      evict.o           lolwut5.c         quicklist.c     release.c         sort.c          zipmap.h
blocked.c     expire.c          lolwut5.o         quicklist.h     release.h         sort.o          zipmap.o
blocked.o     expire.o          lolwut.c          quicklist.o     release.o         sparkline.c     zmalloc.c
childinfo.c   fmacros.h         lolwut.o          rand.c          replication.c     sparkline.h     zmalloc.h
childinfo.o   geo.c             lzf_c.c           rand.h          replication.o     sparkline.o     zmalloc.o
cluster.c     geo.h             lzf_c.o           rand.o          rio.c             stream.h
cluster.h     geohash.c         lzf_d.c           rax.c           rio.h             syncio.c
cluster.o     geohash.h         lzf_d.o           rax.h           rio.o             syncio.o
config.c      geohash_helper.c  lzf.h             rax_malloc.h    scripting.c       testhelp.h
```

6. 将工具安装到系统中。

   make install

测试步骤

步骤1 创建Redis缓存实例。

步骤2 创建3台弹性云服务器（ECS），ECS选择与实例相同可用区、VPC、子网和安全组。

说明

如果是测试单机或主备实例，创建1台ECS即可。

步骤3 在每台ECS上安装**redis-benchmark**。安装步骤参考**下载与安装工具**。

步骤4 每台ECS上执行测试命令。

```
redis-benchmark -h {IP} -p {Port} -a {password} -n {nreqs} -r {randomkeys} -c {connect_number} -d {datasize} -t {command}
```

参数参考值：-c {connect_number}：200，-n {nreqs}：10000000，-r {randomkeys}：1000000，-d {datasize}：32。

- -h表示实例的域名连接地址或IP地址。
- -p表示实例的端口，默认为6379。
- -a表示实例的连接密码，免密连接的实例无需输入-a {password}。
- -t表示执行具体测试命令合集。例如只测试set命令时，使用-t set；如果要测试ping、get、set命令，则使用 -t ping,set,get，命令间使用“,”分隔。
- -c表示客户端连接数。
- -d表示单条数据大小，单位Byte。
- -n表示测试包数量。
- -r表示使用随机key数量。

步骤5 不断调整客户端连接数，执行**步骤4**，得到每秒最大操作数。

版权所有 © 华为云计算技术有限公司

步骤6 取3台测试ECS得到的每秒操作数总和，即为对应规格的性能数据。

如果测试Redis集群，建议每台测试ECS各开启两个benchmark客户端。

说明

- redis-benchmark 测试cluster集群实例时需要加 --cluster 参数，其他实例类型不需要加。
- 如果想对cluster集群的最大连接数进行性能压测，但是压测到1万连接时程序退出，或者报错 Cannot assign requested address。这说明是测试用的ECS本机性能不足，请先检查自己是否只用了1台ECS进行压测。想要对集群压测，建议准备3台ECS，每台ECS起3个redis-benchmark来测试redis实例的最大连接数。

----结束

1.3 使用 memtier_benchmark 工具进行性能测试

针对DCS Redis实例的性能评估，比如测试某种实例规格的set或get在某个高并发场景下的性能，可参考本文的测试方法。

测试工具

memtier_benchmark是Redis Labs推出的一款命令行工具，它能够产生各种各样的流量模式，可以对Memcached和Redis实例进行基准测试。

这个工具提供了丰富的自定义选项和报表功能，通过命令行界面就能够轻松地使用。

了解memtier_benchmark更多详情，请访问**https://github.com/RedisLabs/memtier_benchmark**。

下载与安装工具

以操作系统为CentOS 8.0为例进行安装。

1. 准备工作。
   a. 安装编译所需的工具。
      yum install -y autoconf
      yum install -y automake
      yum install -y make
      yum install -y gcc-c++
      yum install -y git
   b. 启用PowerTools存储库的方法。
      dnf config-manager --set-enabled PowerTools
   c. 安装需要的依赖库。
      yum install -y pcre-devel
      yum install -y zlib-devel
      yum install -y libmemcached-devel
      yum install -y openssl-devel
2. 安装libevent库。
   yum install -y libevent-devel

3. 下载、编译、安装memtier_benchmark的库。
   a. 根目录下创建文件夹，用于下载memtier_benchmark。
      mkdir /env
   b. 下载memtier_benchmark源码。
      cd /env
      git clone https://github.com/RedisLabs/memtier_benchmark.git
   c. 进入源码目录。
      cd memtier_benchmark
   d. 编译生成可执行文件memtier_benchmark。
      autoreconf -ivf
      ./configure
      make
   e. 将工具安装到系统中。
      make install
4. 运行帮助命令查看是否安装成功。
   memtier_benchmark --help

测试步骤

步骤1 创建Redis缓存实例。

步骤2 创建3台弹性云服务器（ECS），ECS选择与实例相同可用区、VPC、子网和安全组。

说明

如果是测试单机或主备实例，创建1台ECS即可。

步骤3 在每台ECS上安装**memtier_benchmark**。安装步骤参考**下载与安装工具**。

步骤4 每台ECS上执行测试命令。

```
memtier_benchmark -s {IP} -n {nreqs}  -c {connect_number}  -t 4 -d {datasize}
```

说明

如果实例类型为cluster集群，则命令为memtier_benchmark --cluster-mode -s {IP} -n {nreqs} -c {connect_number} -t 4 -d {datasize}

参数参考值：-c {connect_number}：200，-n {nreqs}：10000000，-d {datasize}：32。

- -s表示实例的域名连接地址或IP地址。
- -t表示基准测试使用的线程数量
- -c表示客户端连接数
- -d表示单条数据大小，单位byte
- -n表示测试包数量

步骤5 不断调整客户端连接数，执行**步骤4**，得到每秒最大操作数。

步骤6 取3台测试ECS得到的每秒操作数总和，即为对应规格的性能数据。

如果测试Redis集群，建议每台测试ECS各开启两个benchmark客户端。

----结束

测试指标

QPS: 即Query Per Second，表示数据库每秒执行的命令数。

1.4 redis-benchmark 与 memtier_benchmark 的差异

工具	支持Memcached	支持读写比例设置	支持Payload大小随机	支持设置过期时间
memtier_benchmark	是	是	是	是
redis-benchmark	否	否	否	否

2 Redis 3.0 主备实例测试数据

测试环境说明

- 测试实例规格
  Redis 3.0 8G主备
  Redis 3.0 32G主备
- 测试执行机规格
  通用计算增强型 | c6.xlarge.2 | 4vCPUs | 8GB

测试命令

```
redis-benchmark -h {IP} -p {Port} -a {password} -n {nreqs} -r {randomkeys} -c {connection} -d {datasize} -t {command}
```

参数参考值：-c {connect_number}：1000，-n {nreqs}：10000000，-r {randomkeys}：1000000，-d {datasize}：32。

测试结果

表 2-1 SET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)
8G	X86	1000	107657.69	20	23	27
		10000	72750.55	362	366	371
32G	X86	1000	121088.83	9	12	12
		10000	79235.53	203	204	267

版权所有 © 华为云计算技术有限公司

表 2-2 GET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)
8G	X86	1000	119350.25	6	24	27
		10000	77574.7	152	358	365
32G	X86	1000	124650.98	16	17	17
		10000	81991.41	195	196	199

说明

Redis 3.0实例暂不支持ARM CPU类型实例，仅提供X86类型的实例测试结果。

版权所有 © 华为云计算技术有限公司

3 Redis 3.0 Proxy 集群实例测试数据

测试环境说明

- 测试实例规格
  Redis 3.0 64G Proxy集群
- 测试执行机规格
  通用计算增强型 | c6.xlarge.2 | 4vCPUs | 8GB

测试命令

```
redis-benchmark -h {IP} -p {Port} -a {password} -n {nreqs} -r {randomkeys} -c {connection} -d {datasize} -t {command}
```

参数参考值：-c {connect_number}：1000，-n {nreqs}：10000000，-r {randomkeys}：1000000，-d {datasize}：32。

测试结果

表 3-1 SET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)
64G	X86	1000	534960.92	24	34	209
		10000	511362.67	108	171	315

版权所有 © 华为云计算技术有限公司

表 3-2 GET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)
64G	X86	1000	584669.15	23	31	82
		10000	533178.04	144	184	370

说明

Redis 3.0实例暂不支持ARM CPU类型实例，仅提供X86类型的实例测试结果。

版权所有 © 华为云计算技术有限公司

4 Redis 4.0/5.0 主备实例测试数据

测试环境说明

- 测试实例规格
  Redis 4.0/5.0 8G主备
  Redis 4.0/5.0 32G主备
- 测试执行机规格
  通用计算增强型 | c6.2xlarge.2 | 8vCPUs | 16GB
- 测试执行机镜像
  Ubuntu 18.04 server 64bit
- 测试工具
  使用单台ECS测试，测试工具为redis-benchmark

测试命令

```
redis-benchmark -h {IP} -p {Port} -a {password} -n {nreqs} -r {randomkeys} -c {connection} -d {datasize} -t {command}
```

参数参考值：-c {connect_number}：500，-n {nreqs}：10000000，-r {randomkeys}：1000000，-d {datasize}：32，-t {command}：set。

测试结果

说明

- 以下测试结果仅供参考，不同局点环境和网络波动等客观条件可能产生性能差异。
- QPS：即Query Per Second，表示每秒处理的读写操作数，单位是次/秒。
- 平均/最大时延：操作的平均/最大延迟时间，单位为毫秒（ms）。
- x%延迟：指x%操作的延迟时间，单位为毫秒（ms）。例如该指标的值为10ms，99.99%延迟表示99.99%的请求可以在10ms内被处理。

版权所有 © 华为云计算技术有限公司

表 4-1 SET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)	平均时延(ms)
8G	X86	500	132068.98	11	18	205	3.298
		10000	82386.58	171	178	263	69.275
8G	ARM	500	94811.89	10	12	13	3.476
		10000	61264.37	340	350	351	83.848
32G	X86	500	131385.33	9.5	16	17	3.333
		10000	82275.41	157	162.18	162.43	62.105
32G	ARM	500	117553.02	8	21	22	3.875
		10000	76001.7	175	386	387	99.362

表 4-2 GET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)	平均时延(ms)
8G	X86	500	138652.02	7	11	12	2.117
		10000	82710.94	123.7	281.6	282.9	61.078
8G	ARM	500	95432.59	8.8	10	214	3.186
		10000	60984.16	217	337.15	337.92	83.321
32G	X86	500	139113.02	6.6	10	11	2.119
		10000	82489.36	100	105.66	106	60.968

版权所有 © 华为云计算技术有限公司

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)	平均时延(ms)
32G	ARM	500	139041.45	6	10	11	2.487
		10000	81563.41	141	149	150	63

版权所有 © 华为云计算技术有限公司

5 Redis 4.0/5.0 Proxy 实例测试数据

测试环境说明

- 测试实例规格
  Redis 4.0/5.0 64G（8分片） Proxy集群
- 测试执行机规格
  通用计算增强型 | c6.xlarge.2 | 4vCPUs | 8GB
- 测试工具
  使用三台ECS并发测试，测试工具为memtier_benchmark

测试命令

```
memtier_benchmark --ratio=（1:0 and 0:1）-s {IP} -n {nreqs}  -c {connect_number}  -t 4 -d {datasize}
```

参数参考值：-c {connect_number}：1000，-n {nreqs}：10000000，-d {datasize}：32。

测试结果

说明

- 以下测试结果仅供参考，不同局点环境和网络波动等客观条件可能产生性能差异。
- QPS：即Query Per Second，表示每秒处理的读写操作数，单位是次/秒。
- 平均/最大时延：操作的平均/最大延迟时间，单位为毫秒（ms）。
- x%延迟：指x%操作的延迟时间，单位为毫秒（ms）。例如该指标的值为10ms，99.99%延迟表示99.99%的请求可以在10ms内被处理。

表 5-1 SET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	95%左右延迟（ms）	99.99%延迟（ms）	最大时延（ms）
64G	X86	3000	1,323,935.00	3.3	9.4	220
		5000	1,373,756.00	5.3	13	240

版权所有 © 华为云计算技术有限公司

实例规格	实例CPU类型	并发连接数（个）	QPS	95%左右延迟（ms）	99.99%延迟（ms）	最大时延（ms）
		10000	1,332,074.00	11	26	230
		80000	946,032.00	110	460	6800
64G	ARM	3000	837,864.92	5.8	16	78
		5000	763,609.69	10	29	240
		10000	703,808.39	20	47	250
		80000	625,841.69	170	410	940

表 5-2 GET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	95%左右延迟（ms）	99.99%延迟（ms）	最大时延（ms）
64G	X86	3000	1,366,153.00	3.3	9.3	230
		5000	1,458,451.00	5.1	13	220
		10000	1,376,399.00	11	29	440
		80000	953,837.00	120	1300	2200
64G	ARM	3000	764,114.55	6.1	17	100
		5000	765,187.74	10	27	230
		10000	731,310.95	20	47	250
		80000	631,373.33	170	1300	1900

版权所有 © 华为云计算技术有限公司

6 Redis 4.0/5.0 Cluster 集群实例测试数据

测试环境说明

- 测试实例规格
  Redis 4.0/5.0 32G Cluster集群
- 测试执行机规格
  通用计算增强型 | c6.xlarge.2 | 4vCPUs | 8GB
- 测试工具
  使用三台ECS并发测试，测试工具为memtier_benchmark

测试命令

```
memtier_benchmark --cluster-mode --ratio=（1:0 and 0:1）-s {IP} -n {nreqs}  -c {connect_number}  -t 4 -d {datasize}
```

参数参考值：-c {connect_number}：1000，-n {nreqs}：10000000，-d {datasize}：32。

测试结果

说明

- 以下测试结果仅供参考，不同局点环境和网络波动等客观条件可能产生性能差异。
- QPS：即Query Per Second，表示每秒处理的读写操作数，单位是次/秒。
- 平均/最大时延：操作的平均/最大延迟时间，单位为毫秒（ms）。
- x%延迟：指x%操作的延迟时间，单位为毫秒（ms）。例如该指标的值为10ms，99.99%延迟表示99.99%的请求可以在10ms内被处理。

表 6-1 SET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)
32G	X86	1000	371780.2	5.6	6.3	44

版权所有 © 华为云计算技术有限公司

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)
		10000	256073.11	90	220	460
32G	ARM	1000	317053.78	17	34	230
		10000	248832.33	410	490	750

表 6-2 GET 操作命令测试结果

实例规格	实例CPU类型	并发连接数（个）	QPS	99.99%延迟(ms)	第一个100%延迟(ms)	最后一个100%延迟(ms)
32G	X86	1000	427000.04	5.0	5.3	78
		10000	302159.03	63	220	460
32G	ARM	1000	421402.06	13	14	65
		10000	309359.18	180	260	500

版权所有 © 华为云计算技术有限公司

7 Redis 6.0 主备实例测试数据

Redis 6.0基础版实例支持开启SSL，本章节包含开启SSL前后的Redis实例性能测试数据。

测试环境说明

- 测试实例规格
  Redis 6.0 基础版 8G主备
  Redis 6.0 基础版 32G主备
- 测试执行机规格
  通用计算增强型 | 8vCPUs | 16GiB | c7.2xlarge.2
- 测试执行机镜像
  Ubuntu 18.04 server 64bit
- 测试工具
  使用单台ECS测试，测试工具为memtier_benchmark

测试命令

未开启SSL场景：

```
./memtier_benchmark -s {IP} -p {port} -c {connect_number} -t {thread}  -n allkeys --key-prefix="xxxx" --key-minimum=1 --key-maximum={max_key} --key-pattern=P:P --ratio=1:0 -d {datasize}
```

参数参考值：-c {connect_number}：1000，--key-maximum{max_key}：2000000，-d {datasize}：32。

开启SSL场景：

```
./memtier_benchmark -s {IP} -p {port} -c {connect_number} -t {thread}  -n allkeys --key-prefix="xxxx" --key-minimum=1 --key-maximum={max_key} --key-pattern=P:P --ratio=1:0 -d {datasize}  --tls --cacert ca.crt
```

参数参考值：-c {connect_number}：1000，--key-maximum{max_key}：2000000，-d {datasize}：32。

版权所有 © 华为云计算技术有限公司

测试结果

说明

- 以下测试结果仅供参考，不同局点环境和网络波动等客观条件可能产生性能差异。
- QPS：即Query Per Second，表示每秒处理的读写操作数，单位是次/秒。
- 平均/最大时延：操作的平均/最大延迟时间，单位为毫秒（ms）。
- x%延迟：指x%操作的延迟时间，单位为毫秒（ms）。例如该指标的值为10ms，99.99%延迟表示99.99%的请求可以在10ms内被处理。

表 7-1 SET 操作命令测试结果（未开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
8G	X86	500	151047.41	3.355	6.175	12.223
		1000	149346.86	6.673	11.711	31.743
32G	X86	500	143648.1	3.476	5.215	13.055
		4000	104517.03	37.881	139.263	175.103

表 7-2 SET 操作命令测试结果（开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
8G	X86	500	86827.84	5.537	8.575	9.535
		1000	92413.99	10.055	15.615	17.279
32G	X86	500	87385.5	5.584	8.383	9.343
		4000	50813.67	62.623	100.863	104.959

版权所有 © 华为云计算技术有限公司

表 7-3 GET 操作命令测试结果（未开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
8G	X86	500	180413.66	2.764	4.287	11.583
		1000	179113.5	5.586	8.959	29.823
32G	X86	500	175268.86	2.848	4.079	11.839
		4000	134755.17	29.161	126.463	166.911

表 7-4 GET 操作命令测试结果（开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
8G	X86	500	113637.22	4.316	6.239	7.359
		1000	105504.55	8.962	13.439	15.295
32G	X86	500	100309.99	4.603	6.559	6.943
		4000	57007.69	55.052	85.503	89.087

版权所有 © 华为云计算技术有限公司

8 Redis 6.0 Cluster 集群实例测试数据

Redis 6.0基础版实例支持开启SSL，本章节包含开启SSL前后的Redis实例性能测试数据。

测试环境说明

- 测试实例规格
  Redis 6.0 基础版 32G Cluster集群
- 测试执行机规格
  通用计算增强型 | 8vCPUs | 16GiB | c7.2xlarge.2
- 测试执行机镜像
  Ubuntu 18.04 server 64bit
- 测试工具
  使用三台ECS并发测试，测试工具为memtier_benchmark

测试命令

未开启SSL场景：

```
./memtier_benchmark -s {IP} -p {port} -c {connect_number} -t {thread}  -n allkeys --key-prefix="xxxx" --key-minimum=1 --key-maximum={max_key} --key-pattern=P:P --ratio=1:0 -d {datasize} --cluster-mode
```

参数参考值：-c {connect_number}：1000，--key-maximum{max_key}：2000000，-d {datasize}：32。

开启SSL场景：

```
./memtier_benchmark -s {IP} -p {port} -c {connect_number} -t {thread}  -n allkeys --key-prefix="xxxx" --key-minimum=1 --key-maximum={max_key} --key-pattern=P:P --ratio=1:0 -d {datasize} --cluster-mode  --tls --cacert ca.crt
```

参数参考值：-c {connect_number}：1000，--key-maximum{max_key}：2000000，-d {datasize}：32。

版权所有 © 华为云计算技术有限公司

测试结果

说明

- 以下测试结果仅供参考，不同局点环境和网络波动等客观条件可能产生性能差异。
- QPS：即Query Per Second，表示每秒处理的读写操作数，单位是次/秒。
- 平均/最大时延：操作的平均/最大延迟时间，单位为毫秒（ms）。
- x%延迟：指x%操作的延迟时间，单位为毫秒（ms）。例如该指标的值为10ms，99.99%延迟表示99.99%的请求可以在10ms内被处理。

表 8-1 SET 操作命令测试结果（未开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
32G	X86	1000	322899.21	2.661	4.319	8.511
		3000	360336.14	7.757	13.055	29.439
		10000	330378.22	29.411	97.279	153.599

表 8-2 SET 操作命令测试结果（开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
32G	X86	1000	238307.26	3.603	5.151	6.527
		3000	185455.62	13.196	20.607	352.255
		10000	111913.19	57.537	96.767	121.343

表 8-3 GET 操作命令测试结果（未开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
32G	X86	1000	450422.66	1.875	2.767	6.879
		3000	432450.2	6.451	12.095	28.415

版权所有 © 华为云计算技术有限公司

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
		10000	507338.44	23.001	95.231	176.127

表 8-4 GET 操作命令测试结果（开启 SSL 场景）

实例规格	实例CPU类型	并发连接数（个）	QPS	平均时延（ms）	99%延迟（ms）	99.9%延迟（ms）
32G	X86	1000	274066.16	3.076	4.255	7.071
		3000	201063.51	11.743	18.047	387.071
		10000	116026.38	51.284	84.479	136.191

版权所有 © 华为云计算技术有限公司

9 Redis 备份恢复迁移性能测试数据

测试环境说明

- 测试实例规格：
  Redis 5.0 8G主备
  Redis 5.0 32G主备
  Redis 5.0 64G Proxy集群（ 副本数：2 | 分片数：8 | 分片容量：8 GB ）
  Redis 5.0 256G Proxy集群（ 副本数：2 | 分片数：32 | 分片容量：8 GB ）
  Redis 5.0 64G Cluster集群（ 副本数：2 | 分片数：8 | 分片容量：8 GB ）
  Redis 5.0 256G Cluster集群 （ 副本数：2 | 分片数：32 | 分片容量：8 GB ）
- 测试执行机规格：
  c6s.large.2 2vCPUs | 4GB

测试命令

256G proxy实例测试命令：

```
redis-benchmark - h {IP} -p{Port} -n 10000000 -r 10000000 -c 10000 -d 1024
```

256G cluster实例测试命令：

```
redis-benchmark - h {IP} -p{Port} -n 10000000 -r 10000000 -c 40000 -d 1024 -c
```

测试结果

表 9-1 迁移

源实例类型	源实例规格（GB）	目标实例类型	目标实例规格（GB）	迁移方式	数据量（GB）	时间（min）
Redis 5.0 主备	8	Redis 5.0 主备	8	全量迁移+增量迁移	7.78	3

版权所有 © 华为云计算技术有限公司

源实例类型	源实例规格（GB）	目标实例类型	目标实例规格（GB）	迁移方式	数据量（GB）	时间（min）
Redis 5.0主备	32	Redis 5.0主备	32	全量迁移+增量迁移	31.9	17
Redis 5.0 proxy	64	Redis 5.0proxy	64	全量迁移+增量迁移	62.42	7
Redis 5.0 cluster	64	Redis 5.0cluster	64	全量迁移+增量迁移	57.69	6
Redis 5.0 proxy	256	Redis 5.0proxy	256	全量迁移+增量迁移	241.48	23
Redis 5.0 cluster	256	Redis 5.0cluster	256	全量迁移+增量迁移	240.21	22

表 9-2 备份

实例类型	实例规格（GB）	备份方式	数据量（GB）	时间（min）
Redis 5.0主备	8	rdb	7.78	2
Redis 5.0主备	32	rdb	31.9	5
Redis 5.0 proxy	64	rdb	62.42	9
Redis 5.0 proxy	256	rdb	241.48	37
Redis 5.0 cluster	64	rdb	57.69	9
Redis 5.0 cluster	256	rdb	255	39
Redis 5.0主备	8	aof	7.9	2
Redis 5.0主备	32	aof	31.15	10
Redis 5.0 proxy	64	aof	62.42	20
Redis 5.0 proxy	256	aof	241.48	48

版权所有 © 华为云计算技术有限公司

实例类型	实例规格（GB）	备份方式	数据量（GB）	时间（min）
Redis 5.0 cluster	64	aof	57.69	19
Redis 5.0 cluster	256	aof	255	51

表 9-3 恢复

实例类型	实例规格（GB）	恢复方式	数据量（GB）	时间（min）
Redis 5.0主备	8	rdb	7.9	2
Redis 5.0主备	32	rdb	31.15	6
Redis 5.0 proxy	64	rdb	62.42	10
Redis 5.0 proxy	256	rdb	246	42
Redis 5.0 cluster	64	rdb	57.69	10
Redis 5.0 cluster	256	rdb	255	40
Redis 5.0主备	8	aof	7.9	3
Redis 5.0主备	32	aof	31.15	10
Redis 5.0 proxy	64	aof	62.42	10
Redis 5.0 proxy	256	aof	246	46
Redis 5.0 cluster	64	aof	57.69	10
Redis 5.0 cluster	256	aof	255	43

版权所有 © 华为云计算技术有限公司