

**DCS Performance White Paper**

# **DCS Performance White Paper**

**Issue** 01  
**Date** 2024-09-30



**Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2024. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

## **Trademarks and Permissions**



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

## **Notice**

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

## **Huawei Cloud Computing Technologies Co., Ltd.**

Address: Huawei Cloud Data Center Jiaoxinggong Road  
Qianzhong Avenue  
Gui'an New District  
Gui Zhou 550029  
People's Republic of China

Website: <https://www.huaweicloud.com/intl/en-us/>

---

# Contents

---

<b>1 Introduction.....</b>	<b>1</b>
1.1 Testing Redis Performance Using redis-benchmark.....	1
1.2 Testing Redis Performance Using memtier_benchmark.....	5
1.3 Comparing redis-benchmark and memtier_benchmark.....	7
<b>2 Test Data of Master/Standby DCS Redis 3.0 Instances.....</b>	<b>8</b>
<b>3 Test Data of Proxy Cluster DCS Redis 3.0 Instances.....</b>	<b>10</b>
<b>4 Test Data of Master/Standby DCS Redis 4.0 or 5.0 Instances.....</b>	<b>12</b>
<b>5 Test Data of Proxy Cluster DCS Redis 4.0 or 5.0 Instances.....</b>	<b>15</b>
<b>6 Test Data of Redis Cluster DCS Redis 4.0 or 5.0 Instances.....</b>	<b>18</b>
<b>7 Test Data of Master/Standby DCS Redis 6.0 Instances.....</b>	<b>20</b>
<b>8 Test Data of Redis Cluster DCS Redis 6.0 Instances.....</b>	<b>23</b>
<b>9 Test Data of Redis Backup, Restoration, and Migration.....</b>	<b>26</b>

# 1 Introduction

---

## 1.1 Testing Redis Performance Using redis-benchmark

The Redis client includes redis-benchmark, a performance testing utility that simulates N clients concurrently sending M number of query requests.

This section describes how to use redis-benchmark to test the performance of a DCS Redis instance when running command **SET** or **GET** in a high-concurrency scenario.

### Test Procedure

**Step 1** Create a DCS Redis instance.

**Step 2** Create three ECSs and configure the same AZ, VPC, subnet, and security group for the ECSs and the instance.

 **NOTE**

Only one ECS is required for testing on a single-node or master/standby instance.

**Step 3** Install redis-benchmark on each ECS. The Redis server can be installed in either of the following ways and benchmark will be installed, too.

- Method 1:
  - a. Download a Redis client. This example uses redis-6.0.9.  
**wget http://download.redis.io/releases/redis-6.0.9.tar.gz**
  - b. Decompress the client installation package.  
**tar xzf redis-6.0.9.tar.gz**
  - c. Go to the **src** directory of redis-6.0.9.  
**cd redis-6.0.9/src**
  - d. Compile the source code.  
**make**

After the compilation is complete, the tool is stored in the **src** directory of **redis-x.x.x**.

- e. Check whether the redis-benchmark executable file exists.

ls

```
[root@ ~]# 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            mkreleashehdr.sh  redisassert.h  sds.o          t_set.c
ae_export.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  server.c       t_stream.c
ae_kqueue.c   db.c         intset.c         modules       redis-check-aof  server.h        t_string.c
ae.o          db.o         intset.o         multi.c       redis-check-aof.c  server.o       t_string.o
ae_select.c   debug.c      intset.o         multi.o       redis-check-aof.o  setproctitle.c  t_zset.c
anet.c        debugmacro.h  latency.c        networking.c  redis-check-rdb  setproctitle.o  t_zset.o
anet.h        debug.o      latency.h        networking.o  redis-check-rdb.c  sha1.c         util.c
anet.o        defrag.c     lazyfree.c      notify.c      redis-check-rdb.o  sha1.h         util.h
aof.c         defrag.o     lazyfree.o      notify.o      redis-check-rdb.o  sha1.o         util.o
aof.o         dict.c       listpack.c      object.c      redis-cli         siphash.c      valgrind.sup
asciilogo.h   dict.o       listpack.h      pgsort.c     redis-cli.c       siphash.o      version.h
atomicvar.h   dict.o       listpack_malloc.h  pgsort.h     redis-cli.o       siphash.o      version.h
bio.c         endianconv.c  listpack.o      pgsort.o     redis-module.h    slowlog.c      ziplist.c
bio.h         endianconv.h  listpack.o      pgsort.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      localtime.o     pubsub.o     release.c         sort.c         zipmap.o
blocked.c     expire.c     lolwut5.c       quicklist.c  release.h         sort.o         zmalloc.c
blocked.o     expire.o     lolwut5.o       quicklist.h  release.o         sparkline.c    zmalloc.o
childinfo.c   fmacros.h    lolwut.o        quicklist.o  replication.c     sparkline.h    zmalloc.h
childinfo.o   geo.c        lolwut.c        rand.c        replication.o     sparkline.o    zmalloc.o
cluster.c     geo.h        lolwut.o        rand.o        rio.c             stream.h
cluster.h     geohash.c    lzfp.c          rand.h        rio.h             syncio.c
cluster.o     geohash.h    lzfp.o          rax.c         rio.o             syncio.o
config.c      geohash_helper.c  lzfp.o          rax.h         rio.o             syncio.o
config.o      geohash_helper.o  lzfp.o          rax_malloc.h  scripting.c       testhelp.h
```

- f. Install the tool in the system.

make install

- Method 2:

Install the Redis server matching the ECS OS. The following examples use Ubuntu and CentOS.

- Ubuntu

```
sudo apt update
sudo apt install redis-server
```

- CentOS

```
sudo yum install epel-release
sudo yum update
sudo yum -y install redis
```

**Step 4** Run the following test command on all ECSs:

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

Reference values: **-c {connect\_number}: 200; -n {nreqs}: 10,000,000; -r {randomkeys}: 1,000,000; -d {datasize}: 32**

- **-h:** instance domain name or IP address
- **-p:** port of the instance. The default value is **6379**.
- **-a:** password for connecting to the instance. This parameter is not required for password-free instances.
- **-t** Set of commands to be executed For example, to test only the **set** command, use **-t set**. To test the **ping**, **get**, and **set** commands, use **-t ping,set,get**. Use commas (,) to separate commands.
- **-c** number of client connections
- **-d** size of a single data record in bytes
- **-n** number of test packets
- **-r** number of random keys

**Step 5** Repeat **Step 4** with different client connections to obtain the maximum QPS (Query per Second, number of read and write operations per second).

**Step 6** The sum of operations per second of all the three ECSs indicates the performance of the instance specification.

To test on a Redis Cluster instance, launch two benchmark tools on each ECS.

 **NOTE**

- Add the **--cluster** parameter only when testing Redis Cluster instances using redis-benchmark.
- In a test for the maximum number of connections of a Redis Cluster instance, if the performance of the ECSs is insufficient, the program will exit or the error message "Cannot assign requested address" will be displayed when the number of connections reaches 10,000. In this case, check how many ECSs are used in the test. Prepare three ECSs and start three redis-benchmark processes on each ECS.

----End

## Common redis-cli Options

- **-h <hostname>**: host name of the server, which can be an IP address or a domain name.
- **-p <port>**: port of the server. The default value is **6379**.
- **-a <password>**: password for connecting to the server. This parameter is not required for password-free instances.
- **-r <repeat>**: number of times that a command is run.
- **-n <db>**: DB number. The default value is **0**.
- **-c**: cluster mode (with **-ASK** and **-MOVED** redirections).
- **--latency**: a loop where latency is measured continuously.
- **--scan**: scans the key space without blocking the Redis server. (By contrast, scanning using **KEYS \*** blocks Redis server).
- **--eval <file>**: sends the **EVAL** command using a Lua script.
- **-x**: reads the last parameter in stdin.
- **--bigscan**: scans big keys in the data set.
- **--raw**: forces raw data output from the hexadecimal format, such as **\xe4\xb8**.

For more information about redis-cli, visit <https://redis.io/docs/connect/cli/>

## Examples of Common redis-cli Commands

- Connect to an instance:  
**./redis-cli -h {IP} -p 6379**
- Connect to a specified DB:  
**./redis-cli -h {IP} -p 6379 -n 10**
- Connect to a Redis Cluster instance:  
**./redis-cli -h {IP} -p 6379 -c**
- Test the latency (by sending the **ping** command):  
**./redis-cli -h {IP} -p 6379 --latency**
- Scan for keys that match the specified pattern:  
**./redis-cli -h {IP} -p 6379 --scan --pattern '\*:12345\*'**

## Common Options in redis-benchmark (redis-6.0.9)

- **-h <hostname>**: host name of the server, which can be an IP address or a domain name.
- **-p <port>**: port of the server. The default value is **6379**.
- **-a <password>**: password for connecting to the server. This parameter is not required for password-free instances.
- **-c <clients>**: number of concurrent connections. The default value is **50**.
- **-n <requests>**: total number of requests. The default value is **100000**.
- **-d <size>**: data size of the **SET/GET** value, in bytes. The default value is **2**.
- **--dbnum <db >**: database number. The default value is **0**.
- **--threads <num>**: multi-thread mode, which is supported only by redis-benchmark compiled in Redis 6.0. In pressure tests, the multi-thread mode outperforms the single-thread mode.
- **--cluster**: cluster mode (required only by Redis Cluster).
- **-k <boolean>**: **1**=keep alive; **0**=reconnect. The default value is **1**, indicating that both pconnect and connect can be tested.
- **-r <keyspacelen>**: uses random keys for **SET**, **GET**, and **INCR**, and random values for **SADD**. *keyspacelen* indicates the number of keys to be added.
- **-e**: displays server errors to stdout.
- **-q**: displays only the number of queries per second.
- **-l**: runs tests in loops.
- **-t <tests>**: tests specified commands.
- **-I**: idle mode. Open *N* idle connections and wait.
- **-P <numreq>**: concurrent pipeline requests. The default value is **1**.

For more information about redis-benchmark, visit [https://redis.io/docs/latest/operate/oss\\_and\\_stack/management/optimization/benchmarks/](https://redis.io/docs/latest/operate/oss_and_stack/management/optimization/benchmarks/).

## Examples of Common redis-benchmark Commands

- Test single-node, master/standby, read/write splitting, and Proxy Cluster instances:  

```
./redis-benchmark -h {IP address or domain name} -p 6379 -a {password} --threads {num} -n {nreqs} -r {randomkeys} -c {clients} -d {datasize} -t {command}
```
- Test Redis Cluster instances:  

```
./redis-benchmark -h {IP address or domain name} -p 6379 -a {password} --threads {num} -n {nreqs} -r {randomkeys} -c {clients} -d {datasize} --cluster -t {command}
```
- Test connect:  

```
./redis-benchmark -h {IP address or domain name} -p 6379 -a {password} --threads {num} -n {nreqs} -r {randomkeys} -c {clients} -d {datasize} -k 0 -t {command}
```
- Test idle connections:  

```
./redis-benchmark -h {IP address or domain name} -p 6379 -a {pwd} -c {clients} -l
```

## 1.2 Testing Redis Performance Using memtier\_benchmark

memtier\_benchmark is a command-line tool developed by Redis Labs. It can generate traffic in various modes and supports Redis. This tool provides multiple options and reporting features that can be easily used through the CLI. For details, visit [https://github.com/RedisLabs/memtier\\_benchmark](https://github.com/RedisLabs/memtier_benchmark).

This section describes how to use memtier-benchmark to test the performance of a DCS Redis instance when running command **SET** or **GET** in a high-concurrency scenario.

### Test Procedure

**Step 1** Create a DCS Redis instance.

**Step 2** Create three ECSs and configure the same AZ, VPC, subnet, and security group for the ECSs and the instance.

 **NOTE**

Only one ECS is required for testing on a single-node or master/standby instance.

**Step 3** Install [memtier\\_benchmark](#) on each ECS. CentOS 8.0 is used as an example.

1. Preparations

a. Install the tools required for compilation.

```
yum install -y autoconf
yum install -y automake
yum install -y make
yum install -y gcc-c++
yum install -y git
```

b. Enable the PowerTools repository.

```
dnf config-manager --set-enabled PowerTools
```

c. Install the required dependencies.

```
yum install -y pcre-devel
yum install -y zlib-devel
yum install -y libmemcached-devel
yum install -y openssl-devel
```

2. Install the libevent library.

```
yum install -y libevent-devel
```

3. Download, compile, and install the memtier\_benchmark library.

a. Create a folder in the root directory where the memtier\_benchmark library will be stored.

```
mkdir /env
```

b. Download the memtier\_benchmark source code.



- ```
cd /env
git clone https://github.com/RedisLabs/memtier_benchmark.git
```
- c. Go to the directory where the source code is located.

```
cd memtier_benchmark
```
  - d. Compile the source code and generate the executable file **memtier\_benchmark**.

```
autoreconf -ivf
./configure
make
```
  - e. Install the tool in the system.

```
make install
```
4. Run the following command to check whether the installation is successful: If a parameter description of **memtier\_benchmark** is returned, the installation is successful.

```
memtier_benchmark --help
```

**Step 4** Run the following test command on all ECSs:

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

 **NOTE**

Run **memtier\_benchmark --cluster-mode -s {IP} -n {nreqs} -c {connect\_number} -t 4 -d {datasize}** for a Redis Cluster instance.

Reference values: **-c {connect\_number}: 200; -n {nreqs}: 10,000,000; -d {datasize}: 32**

- **-s** indicates the domain name or IP address of the instance.
- **-t** indicates the number of threads used in the benchmark test.
- **-c** indicates the number of client connections.
- **-d** indicates the size of a single data record in bytes.
- **-n** indicates the number of test packets.

**Step 5** Repeat **Step 4** with different client connections to obtain the maximum QPS (Query per Second, number of read and write operations per second).

**Step 6** The sum of operations per second of all the three ECSs indicates the performance of the instance specification.

To test on a Redis Cluster instance, launch two benchmark tools on each ECS.

----End

## 1.3 Comparing redis-benchmark and memtier\_benchmark

| Tool              | Memcached     | Setting Read/Write Ratio | Random Payload | Setting Timeout |
|-------------------|---------------|--------------------------|----------------|-----------------|
| memtier_benchmark | Supported     | Supported                | Supported      | Supported       |
| redis-benchmark   | Not supported | Not supported            | Not supported  | Not supported   |

# 2 Test Data of Master/Standby DCS Redis 3.0 Instances

## Test Environment

- Redis instance specifications  
Redis 3.0 | 8 GB | master/standby  
Redis 3.0 | 32 GB | master/standby
- ECS flavors  
General computing-enhanced | c6.xlarge.2 | 4 vCPUs | 8 GB

## Test Command

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

Reference values: **-c {connect\_number}: 1000; -n {nreqs}: 10,000,000; -r {randomkeys}: 1,000,000; -d {datasize}: 32**

## Test Result

Table 2-1 Test result of running the SET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99th-Percentile Latency (ms) | First 100th-Percentile Latency (ms) | Last 100th-Percentile Latency (ms) |
|------------------|----------|------------------------|------------|---------------------------------|-------------------------------------|------------------------------------|
| 8 GB             | x86      | 1000                   | 107,657.69 | 20                              | 23                                  | 27                                 |
|                  |          | 10,000                 | 72,750.55  | 362                             | 366                                 | 371                                |
| 32 GB            | x86      | 1000                   | 121,088.83 | 9                               | 12                                  | 12                                 |
|                  |          | 10,000                 | 79,235.53  | 203                             | 204                                 | 267                                |

**Table 2-2** Test result of running the GET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99 <sup>th</sup> -Percentile Latency (ms) | First 100 <sup>th</sup> -Percentile Latency (ms) | Last 100 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------------------------------|--------------------------------------------------|-------------------------------------------------|
| 8 GB             | x86      | 1000                   | 119,350.25 | 6                                            | 24                                               | 27                                              |
|                  |          | 10,000                 | 77,574.7   | 152                                          | 358                                              | 365                                             |
| 32 GB            | x86      | 1000                   | 124,650.98 | 16                                           | 17                                               | 17                                              |
|                  |          | 10,000                 | 81,991.41  | 195                                          | 196                                              | 199                                             |

 **NOTE**

DCS for Redis 3.0 does not support the Arm CPU architecture, so only x86-based instance test results are provided.

# 3 Test Data of Proxy Cluster DCS Redis 3.0 Instances

## Test Environment

- Redis instance specifications  
Redis 3.0 | 64 GB | Proxy Cluster
- ECS flavors  
General computing-enhanced | c6.xlarge.2 | 4 vCPUs | 8 GB

## Test Command

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

Reference values: **-c {connect\_number}: 1000; -n {nreqs}: 10,000,000; -r {randomkeys}: 1,000,000; -d {datasize}: 32**

## Test Result

Table 3-1 Test result of running the SET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99th-Percentile Latency (ms) | First 100th-Percentile Latency (ms) | Last 100th-Percentile Latency (ms) |
|------------------|----------|------------------------|------------|---------------------------------|-------------------------------------|------------------------------------|
| 64 GB            | x86      | 1000                   | 534,960.92 | 24                              | 34                                  | 209                                |
|                  |          | 10,000                 | 511,362.67 | 108                             | 171                                 | 315                                |

**Table 3-2** Test result of running the GET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99 <sup>th</sup> -Percentile Latency (ms) | First 100 <sup>th</sup> -Percentile Latency (ms) | Last 100 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------------------------------|--------------------------------------------------|-------------------------------------------------|
| 64 GB            | x86      | 1000                   | 584,669.15 | 23                                           | 31                                               | 82                                              |
|                  |          | 10,000                 | 533,178.04 | 144                                          | 184                                              | 370                                             |

 **NOTE**

DCS for Redis 3.0 does not support the Arm CPU architecture, so only x86-based instance test results are provided.

# 4 Test Data of Master/Standby DCS Redis 4.0 or 5.0 Instances

## Test Environment

- Redis instance specifications
  - Redis 4.0 or 5.0 | 8 GB | master/standby
  - Redis 4.0 or 5.0 | 32 GB | master/standby
- ECS flavors
  - General computing-enhanced | c6.2xlarge.2 | 8 vCPUs | 16 GB
- ECS image
  - Ubuntu 18.04 server 64-bit
- Test tool
  - A single ECS is used for the test. The test tool is redis-benchmark.

## Test Command

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

Reference values: **-c {connect\_number}: 500; -n {nreqs}: 10,000,000; -r {randomkeys}: 1,000,000; -d {datasize}: 32; -t {command}: set**

## Test Result

### NOTE

- The following test results are for reference only. The performance may vary depending on the site environment and network fluctuation.
- QPS: Query per second, indicates number of read and write operations per second. Unit: count/second.
- Average Latency: Average latency of operations, in milliseconds.
- x<sup>th</sup> Percentile Latency: latency of x% of operations, in milliseconds. For example, if the value is 10 ms, 99.99<sup>th</sup> percentile latency indicates that 99.99% queries can be processed within 10 ms.

**Table 4-1** Test result of running the SET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99 <sup>th</sup> -Percentile Latency (ms) | First 100 <sup>th</sup> -Percentile Latency (ms) | Last 100 <sup>th</sup> -Percentile Latency (ms) | Average Latency (ms) |
|------------------|----------|------------------------|------------|----------------------------------------------|--------------------------------------------------|-------------------------------------------------|----------------------|
| 8 GB             | x86      | 500                    | 132,068.98 | 11                                           | 18                                               | 205                                             | 3.298                |
|                  |          | 10,000                 | 82,386.58  | 171                                          | 178                                              | 263                                             | 69.275               |
| 8 GB             | Arm      | 500                    | 94,811.89  | 10                                           | 12                                               | 13                                              | 3.476                |
|                  |          | 10,000                 | 61,264.37  | 340                                          | 350                                              | 351                                             | 83.848               |
| 32 GB            | x86      | 500                    | 131,385.33 | 9.5                                          | 16                                               | 17                                              | 3.333                |
|                  |          | 10,000                 | 82,275.41  | 157                                          | 162.18                                           | 162.43                                          | 62.105               |
| 32 GB            | Arm      | 500                    | 117,553.02 | 8                                            | 21                                               | 22                                              | 3.875                |
|                  |          | 10,000                 | 76,001.7   | 175                                          | 386                                              | 387                                             | 99.362               |

**Table 4-2** Test result of running the GET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99 <sup>th</sup> -Percentile Latency (ms) | First 100 <sup>th</sup> -Percentile Latency (ms) | Last 100 <sup>th</sup> -Percentile Latency (ms) | Average Latency (ms) |
|------------------|----------|------------------------|------------|----------------------------------------------|--------------------------------------------------|-------------------------------------------------|----------------------|
| 8 GB             | x86      | 500                    | 138,652.02 | 7                                            | 11                                               | 12                                              | 2.117                |
|                  |          | 10,000                 | 82,710.94  | 123.7                                        | 281.6                                            | 282.9                                           | 61.078               |
| 8 GB             | Arm      | 500                    | 95,432.59  | 8.8                                          | 10                                               | 214                                             | 3.186                |
|                  |          | 10,000                 | 60,984.16  | 217                                          | 337.15                                           | 337.92                                          | 83.321               |
| 32 GB            | x86      | 500                    | 139,113.02 | 6.6                                          | 10                                               | 11                                              | 2.119                |



| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99 <sup>th</sup> -Percentile Latency (ms) | First 100 <sup>th</sup> -Percentile Latency (ms) | Last 100 <sup>th</sup> -Percentile Latency (ms) | Average Latency (ms) |
|------------------|----------|------------------------|------------|----------------------------------------------|--------------------------------------------------|-------------------------------------------------|----------------------|
|                  |          | 10,000                 | 82,489.36  | 100                                          | 105.66                                           | 106                                             | 60.968               |
| 32 GB            | Arm      | 500                    | 139,041.45 | 6                                            | 10                                               | 11                                              | 2.487                |
|                  |          | 10,000                 | 81,563.41  | 141                                          | 149                                              | 150                                             | 63                   |

# 5 Test Data of Proxy Cluster DCS Redis 4.0 or 5.0 Instances

## Test Environment

- Redis instance specifications  
Redis 4.0 or 5.0 | 64 GB | 8 shards | Proxy Cluster
- ECS flavors  
General computing-enhanced | c6.xlarge.2 | 4 vCPUs | 8 GB
- Test tool  
Three ECSs are used for concurrent tests. The test tool is memtier\_benchmark.

## Test Command

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

Reference values: **-c {connect\_number}: 1000; -n {nreqs}: 10,000,000; -d {datasize}: 32**

## Test Result

### NOTE

- The following test results are for reference only. The performance may vary depending on the site environment and network fluctuation.
- QPS: Query per second, indicates number of read and write operations per second. Unit: count/second.
- Average Latency: Average latency of operations, in milliseconds.
- x<sup>th</sup> Percentile Latency: latency of x% of operations, in milliseconds. For example, if the value is 10 ms, 99.99<sup>th</sup> percentile latency indicates that 99.99% queries can be processed within 10 ms.

**Table 5-1** Test result of running the SET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS          | 95 <sup>th</sup> -Percentile Latency (ms) | 99.99 <sup>th</sup> -Percentile Latency (ms) | Maximum Latency (ms) |
|------------------|----------|------------------------|--------------|-------------------------------------------|----------------------------------------------|----------------------|
| 64 GB            | x86      | 3000                   | 1,323,935.00 | 3.3                                       | 9.4                                          | 220                  |
|                  |          | 5000                   | 1,373,756.00 | 5.3                                       | 13                                           | 240                  |
|                  |          | 10,000                 | 1,332,074.00 | 11                                        | 26                                           | 230                  |
|                  |          | 80,000                 | 946,032.00   | 110                                       | 460                                          | 6800                 |
| 64 GB            | Arm      | 3000                   | 837,864.92   | 5.8                                       | 16                                           | 78                   |
|                  |          | 5000                   | 763,609.69   | 10                                        | 29                                           | 240                  |
|                  |          | 10,000                 | 703,808.39   | 20                                        | 47                                           | 250                  |
|                  |          | 80,000                 | 625,841.69   | 170                                       | 410                                          | 940                  |

**Table 5-2** Test result of running the GET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS          | 95 <sup>th</sup> -Percentile Latency (ms) | 99.99 <sup>th</sup> -Percentile Latency (ms) | Maximum Latency (ms) |
|------------------|----------|------------------------|--------------|-------------------------------------------|----------------------------------------------|----------------------|
| 64 GB            | x86      | 3000                   | 1,366,153.00 | 3.3                                       | 9.3                                          | 230                  |
|                  |          | 5000                   | 1,458,451.00 | 5.1                                       | 13                                           | 220                  |
|                  |          | 10,000                 | 1,376,399.00 | 11                                        | 29                                           | 440                  |
|                  |          | 80,000                 | 953,837.00   | 120                                       | 1300                                         | 2200                 |
| 64 GB            | Arm      | 3000                   | 764,114.55   | 6.1                                       | 17                                           | 100                  |
|                  |          | 5000                   | 765,187.74   | 10                                        | 27                                           | 230                  |

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 95 <sup>th</sup> -Percentile Latency (ms) | 99.99 <sup>th</sup> -Percentile Latency (ms) | Maximum Latency (ms) |
|------------------|----------|------------------------|------------|-------------------------------------------|----------------------------------------------|----------------------|
|                  |          | 10,000                 | 731,310.95 | 20                                        | 47                                           | 250                  |
|                  |          | 80,000                 | 631,373.33 | 170                                       | 1300                                         | 1900                 |

# 6 Test Data of Redis Cluster DCS Redis 4.0 or 5.0 Instances

## Test Environment

- Redis instance specifications  
Redis 4.0 or 5.0 | 32 GB | Redis Cluster
- ECS flavors  
General computing-enhanced | c6.xlarge.2 | 4 vCPUs | 8 GB
- Test tool  
Three ECSs are used for concurrent tests. The test tool is memtier\_benchmark.

## Test Command

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

Reference values: **-c {connect\_number}: 1000; -n {nreqs}: 10,000,000; -d {datasize}: 32**

## Test Result

### NOTE

- The following test results are for reference only. The performance may vary depending on the site environment and network fluctuation.
- QPS: Query per second, indicates number of read and write operations per second. Unit: count/second.
- Average Latency: Average latency of operations, in milliseconds.
- x<sup>th</sup> Percentile Latency: latency of x% of operations, in milliseconds. For example, if the value is 10 ms, 99.99<sup>th</sup> percentile latency indicates that 99.99% queries can be processed within 10 ms.

**Table 6-1** Test result of running the SET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99 <sup>th</sup> -Percentile Latency (ms) | First 100 <sup>th</sup> -Percentile Latency (ms) | Last 100 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------------------------------|--------------------------------------------------|-------------------------------------------------|
| 32 GB            | x86      | 1000                   | 371,780.2  | 5.6                                          | 6.3                                              | 44                                              |
|                  |          | 10,000                 | 256,073.11 | 90                                           | 220                                              | 460                                             |
| 32 GB            | Arm      | 1000                   | 317,053.78 | 17                                           | 34                                               | 230                                             |
|                  |          | 10,000                 | 248,832.33 | 410                                          | 490                                              | 750                                             |

**Table 6-2** Test result of running the GET command

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | 99.99 <sup>th</sup> -Percentile Latency (ms) | First 100 <sup>th</sup> -Percentile Latency (ms) | Last 100 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------------------------------|--------------------------------------------------|-------------------------------------------------|
| 32 GB            | x86      | 1000                   | 427,000.04 | 5.0                                          | 5.3                                              | 78                                              |
|                  |          | 10,000                 | 302,159.03 | 63                                           | 220                                              | 460                                             |
| 32 GB            | Arm      | 1000                   | 421,402.06 | 13                                           | 14                                               | 65                                              |
|                  |          | 10,000                 | 309,359.18 | 180                                          | 260                                              | 500                                             |

# 7 Test Data of Master/Standby DCS Redis 6.0 Instances

DCS Redis 6.0 basic edition instances support SSL. This section covers the performance tested with and without SSL enabled.

## Test Environment

- Redis instance specifications
  - Redis 6.0 | Basic edition | 8 GB | Master/Standby
  - Redis 6.0 | Basic edition | 32 GB | Master/Standby
- ECS flavors
  - General compute-plus | 8 vCPUs | 16 GiB | c7.2xlarge.2
- ECS image
  - Ubuntu 18.04 server 64-bit
- Test tool
  - A single ECS is used for the test. The test tool is memtier\_benchmark.

## Test Command

SSL disabled:

```
./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}
```

Reference values: **-c {connect\_number}: 1000, --key-maximum{max\_key}: 2000000, -d {datasize}: 32**

SSL enabled:

```
./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
```

Reference values: **-c {connect\_number}: 1000, --key-maximum{max\_key}: 2000000, -d {datasize}: 32**

## Test Result

### NOTE

- The following test results are for reference only. The performance may vary depending on the site environment and network fluctuation.
- QPS: Query per second, indicates number of read and write operations per second. Unit: count/second.
- Average Latency: Average latency of operations, in milliseconds.
- x<sup>th</sup> Percentile Latency: latency of x% of operations, in milliseconds. For example, if the value is 10 ms, 99.99<sup>th</sup> percentile latency indicates that 99.99% queries can be processed within 10 ms.

**Table 7-1** Test result of the SET command (SSL disabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | Average Latency (ms) | 99 <sup>th</sup> -Percentile Latency (ms) | 99.9 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------|-------------------------------------------|---------------------------------------------|
| 8 GB             | x86      | 500                    | 151,047.41 | 3.355                | 6.175                                     | 12.223                                      |
|                  |          | 1000                   | 149,346.86 | 6.673                | 11.711                                    | 31.743                                      |
| 32 GB            | x86      | 500                    | 143,648.1  | 3.476                | 5.215                                     | 13.055                                      |
|                  |          | 4000                   | 104,517.03 | 37.881               | 139.263                                   | 175.103                                     |

**Table 7-2** Test result of the SET command (SSL enabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS       | Average Latency (ms) | 99 <sup>th</sup> -Percentile Latency (ms) | 99.9 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|-----------|----------------------|-------------------------------------------|---------------------------------------------|
| 8 GB             | x86      | 500                    | 86,827.84 | 5.537                | 8.575                                     | 9.535                                       |
|                  |          | 1000                   | 92,413.99 | 10.055               | 15.615                                    | 17.279                                      |
| 32 GB            | x86      | 500                    | 87,385.5  | 5.584                | 8.383                                     | 9.343                                       |
|                  |          | 4000                   | 50,813.67 | 62.623               | 100.863                                   | 104.959                                     |



**Table 7-3** Test result of the GET command (SSL disabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | Average Latency (ms) | 99th-Percentile Latency (ms) | 99.9th-Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------|------------------------------|--------------------------------|
| 8 GB             | x86      | 500                    | 180,413.66 | 2.764                | 4.287                        | 11.583                         |
|                  |          | 1000                   | 179,113.5  | 5.586                | 8.959                        | 29.823                         |
| 32 GB            | x86      | 500                    | 175,268.86 | 2.848                | 4.079                        | 11.839                         |
|                  |          | 4000                   | 134,755.17 | 29.161               | 126.463                      | 166.911                        |

**Table 7-4** Test result of the GET command (SSL enabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | Average Latency (ms) | 99th-Percentile Latency (ms) | 99.9th-Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------|------------------------------|--------------------------------|
| 8 GB             | x86      | 500                    | 113,637.22 | 4.316                | 6.239                        | 7.359                          |
|                  |          | 1000                   | 105,504.55 | 8.962                | 13.439                       | 15.295                         |
| 32 GB            | x86      | 500                    | 100,309.99 | 4.603                | 6.559                        | 6.943                          |
|                  |          | 4000                   | 57,007.69  | 55.052               | 85.503                       | 89.087                         |

# 8 Test Data of Redis Cluster DCS Redis 6.0 Instances

---

DCS Redis 6.0 basic edition instances support SSL. This section covers the performance tested with and without SSL enabled.

## Test Environment

- Redis instance specifications  
Redis 6.0 | Basic edition | 32 GB | Redis Cluster
- ECS flavors  
General compute-plus | 8 vCPUs | 16 GiB | c7.2xlarge.2
- ECS image  
Ubuntu 18.04 server 64-bit
- Test tool  
Three ECSs are used for concurrent tests. The test tool is memtier\_benchmark.

## Test Command

SSL disabled:

```
./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
```

Reference values: **-c {connect\_number}: 1000, --key-maximum{max\_key}: 2000000, -d {datasize}: 32**

SSL enabled:

```
./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
```

Reference values: **-c {connect\_number}: 1000, --key-maximum{max\_key}: 2000000, -d {datasize}: 32**

## Test Result

### NOTE

- The following test results are for reference only. The performance may vary depending on the site environment and network fluctuation.
- QPS: Query per second, indicates number of read and write operations per second. Unit: count/second.
- Average Latency: Average latency of operations, in milliseconds.
- x<sup>th</sup> Percentile Latency: latency of x% of operations, in milliseconds. For example, if the value is 10 ms, 99.99<sup>th</sup> percentile latency indicates that 99.99% queries can be processed within 10 ms.

**Table 8-1** Test result of the SET command (SSL disabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | Average Latency (ms) | 99 <sup>th</sup> -Percentile Latency (ms) | 99.9 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------|-------------------------------------------|---------------------------------------------|
| 32 GB            | x86      | 1000                   | 322,899.21 | 2.661                | 4.319                                     | 8.511                                       |
|                  |          | 3000                   | 360,336.14 | 7.757                | 13.055                                    | 29.439                                      |
|                  |          | 10,000                 | 330,378.22 | 29.411               | 97.279                                    | 153,599                                     |

**Table 8-2** Test result of the SET command (SSL enabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | Average Latency (ms) | 99 <sup>th</sup> -Percentile Latency (ms) | 99.9 <sup>th</sup> -Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------|-------------------------------------------|---------------------------------------------|
| 32 GB            | x86      | 1000                   | 238,307.26 | 3.603                | 5.151                                     | 6.527                                       |
|                  |          | 3000                   | 185,455.62 | 13.196               | 20.607                                    | 352.255                                     |
|                  |          | 10,000                 | 111,913.19 | 57.537               | 96.767                                    | 121.343                                     |

**Table 8-3** Test result of the GET command (SSL disabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | Average Latency (ms) | 99th-Percentile Latency (ms) | 99.9th-Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------|------------------------------|--------------------------------|
| 32 GB            | x86      | 1000                   | 450,422.66 | 1.875                | 2.767                        | 6.879                          |
|                  |          | 3000                   | 432,450.2  | 6.451                | 12.095                       | 28.415                         |
|                  |          | 10,000                 | 507,338.44 | 23.001               | 95.231                       | 176.127                        |

**Table 8-4** Test result of the GET command (SSL enabled)

| Redis Cache Size | CPU Type | Concurrent Connections | QPS        | Average Latency (ms) | 99th-Percentile Latency (ms) | 99.9th-Percentile Latency (ms) |
|------------------|----------|------------------------|------------|----------------------|------------------------------|--------------------------------|
| 32 GB            | x86      | 1000                   | 274,066.16 | 3.076                | 4.255                        | 7.071                          |
|                  |          | 3000                   | 201,063.51 | 11.743               | 18.047                       | 387.071                        |
|                  |          | 10,000                 | 116,026.38 | 51.284               | 84.479                       | 136.191                        |

# 9 Test Data of Redis Backup, Restoration, and Migration

---

## Test Environment

- Redis instance specifications
  - Redis 5.0 | 8 GB | master/standby
  - Redis 5.0 | 32 GB | master/standby
  - Redis 5.0 | 64 GB | Proxy Cluster (2 replicas | 8 shards | 8 GB per shard)
  - Redis 5.0 | 256 GB | Proxy Cluster (2 replicas | 32 shards | 8 GB per shard)
  - Redis 5.0 | 64 GB | Redis Cluster (2 replicas | 8 shards | 8 GB per shard)
  - Redis 5.0 | 256 GB | Redis Cluster (2 replicas | 32 shards | 8 GB per shard)
- ECS flavors
  - c6s.large.2 2 vCPUs | 4 GB

## Test Command

Run the following command on a 256 GB Proxy Cluster instance:

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

Run the following command on a 256 GB Redis Cluster instance:

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

## Test Result

**Table 9-1** Migration

| Source Instance Type       | Source Instance Specifications (GB) | Target Instance Type       | Target Instance Specifications (GB) | Migration Type     | Data Volume (GB) | Duration (min) |
|----------------------------|-------------------------------------|----------------------------|-------------------------------------|--------------------|------------------|----------------|
| Redis 5.0   master/standby | 8                                   | Redis 5.0   master/standby | 8                                   | Full + incremental | 7.78             | 3              |
| Redis 5.0   master/standby | 32                                  | Redis 5.0   master/standby | 32                                  | Full + incremental | 31.9             | 17             |
| Redis 5.0   Proxy Cluster  | 64                                  | Redis 5.0   Proxy Cluster  | 64                                  | Full + incremental | 62.42            | 7              |
| Redis 5.0   Redis Cluster  | 64                                  | Redis 5.0   Redis Cluster  | 64                                  | Full + incremental | 57.69            | 6              |
| Redis 5.0   Proxy Cluster  | 256                                 | Redis 5.0   Proxy Cluster  | 256                                 | Full + incremental | 241.48           | 23             |
| Redis 5.0   Redis Cluster  | 256                                 | Redis 5.0   Redis Cluster  | 256                                 | Full + incremental | 240.21           | 22             |

**Table 9-2** Backup

| Instance Type              | Instance Specifications (GB) | Backup Mode | Data Volume (GB) | Duration (min) |
|----------------------------|------------------------------|-------------|------------------|----------------|
| Redis 5.0   master/standby | 8                            | RDB         | 7.78             | 2              |
| Redis 5.0   master/standby | 32                           | RDB         | 31.9             | 5              |
| Redis 5.0   Proxy Cluster  | 64                           | RDB         | 62.42            | 9              |
| Redis 5.0   Proxy Cluster  | 256                          | RDB         | 241.48           | 37             |

| Instance Type              | Instance Specifications (GB) | Backup Mode | Data Volume (GB) | Duration (min) |
|----------------------------|------------------------------|-------------|------------------|----------------|
| Redis 5.0   Redis Cluster  | 64                           | RDB         | 57.69            | 9              |
| Redis 5.0   Redis Cluster  | 256                          | RDB         | 255              | 39             |
| Redis 5.0   master/standby | 8                            | AOF         | 7.9              | 2              |
| Redis 5.0   master/standby | 32                           | AOF         | 31.15            | 10             |
| Redis 5.0   Proxy Cluster  | 64                           | AOF         | 62.42            | 20             |
| Redis 5.0   Proxy Cluster  | 256                          | AOF         | 241.48           | 48             |
| Redis 5.0   Redis Cluster  | 64                           | AOF         | 57.69            | 19             |
| Redis 5.0   Redis Cluster  | 256                          | AOF         | 255              | 51             |

**Table 9-3** Restoration

| Instance Type              | Instance Specifications (GB) | Restoration Mode | Data Volume (GB) | Duration (min) |
|----------------------------|------------------------------|------------------|------------------|----------------|
| Redis 5.0   master/standby | 8                            | RDB              | 7.9              | 2              |
| Redis 5.0   master/standby | 32                           | RDB              | 31.15            | 6              |
| Redis 5.0   Proxy Cluster  | 64                           | RDB              | 62.42            | 10             |
| Redis 5.0   Proxy Cluster  | 256                          | RDB              | 246              | 42             |
| Redis 5.0   Redis Cluster  | 64                           | RDB              | 57.69            | 10             |

| Instance Type              | Instance Specifications (GB) | Restoration Mode | Data Volume (GB) | Duration (min) |
|----------------------------|------------------------------|------------------|------------------|----------------|
| Redis 5.0   Redis Cluster  | 256                          | RDB              | 255              | 40             |
| Redis 5.0   master/standby | 8                            | AOF              | 7.9              | 3              |
| Redis 5.0   master/standby | 32                           | AOF              | 31.15            | 10             |
| Redis 5.0   Proxy Cluster  | 64                           | AOF              | 62.42            | 10             |
| Redis 5.0   Proxy Cluster  | 256                          | AOF              | 246              | 46             |
| Redis 5.0   Redis Cluster  | 64                           | AOF              | 57.69            | 10             |
| Redis 5.0   Redis Cluster  | 256                          | AOF              | 255              | 43             |