



2020 级

《物联网数据存储与管理》课程  
**实 验 报 告**

姓 名 胡鹏飞

学 号 U202015514

班 号 计算机 2007 班

日 期 2023.05.16



# 目 录

一、实验目的 .....	1
二、实验背景 .....	1
三、实验环境 .....	1
四、实验内容 .....	2
4.1 对象存储技术实践 .....	2
4.2 对象存储性能分析 .....	2
五、实验过程 .....	2
5.1 对象存储技术实践 .....	2
5.2 对象存储性能分析 .....	5
六、实验总结 .....	10
参考文献 .....	11

## 一、实验目的

1. 熟悉对象存储技术，代表性系统及其特性；
2. 实践对象存储系统，部署实验环境，进行初步测试；
3. 基于对象存储系统，分析性能问题，架设应用实践。

## 二、实验背景

对象存储，也叫做基于对象的存储，是用来描述解决和处理离散单元的方法的通用术语，这些离散单元被称作为对象。就像文件一样，对象包含数据，但是和文件不同的是，对象在一个层结构中不会再有层级结构。每个对象都在一个被称作存储池的扁平地址空间的同一级别里，一个对象不会属于另一个对象的下一级。

文件和对象都有与它们所包含的数据相关的元数据，但是对象是以扩展元数据为特征的。每个对象都被分配一个唯一的标识符，允许一个服务器或者最终用户来检索对象，而不必知道数据的物理地址。这种方法对于在云计算环境中自动化和简化数据存储有帮助。

对象存储根本上改变了存储蓝图。它处理和解决了曾经被认为是棘手的存储问题：不间断可扩展性、弹性下降、限制数据持久性、无限技术更新和成本失控。

MinIO 是在 GNU Affero 通用公共许可证 v3.0 下发布的高性能对象存储。它是与 Amazon S3 云存储服务兼容的 API。使用 MinIO 为机器学习、分析和应用程序数据工作负载构建高性能基础架构。

## 三、实验环境

硬件环境	CPU	Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz
	内存	16.0 GB
软件环境	操作系统	Windows 11
	其它软件	minio mc s3bench

## 四、实验内容

### 4.1 对象存储技术实践

1. 了解代码管理工具 git，使用 git 进行作业提交
2. 运用 minio 和 mc 创建服务端和客户端，访问和管理数据。
3. 使用 s3bench 进行测试。

### 4.2 对象存储性能分析

修改 s3bench 参数，观察运行情况，绘制图表进行分析。

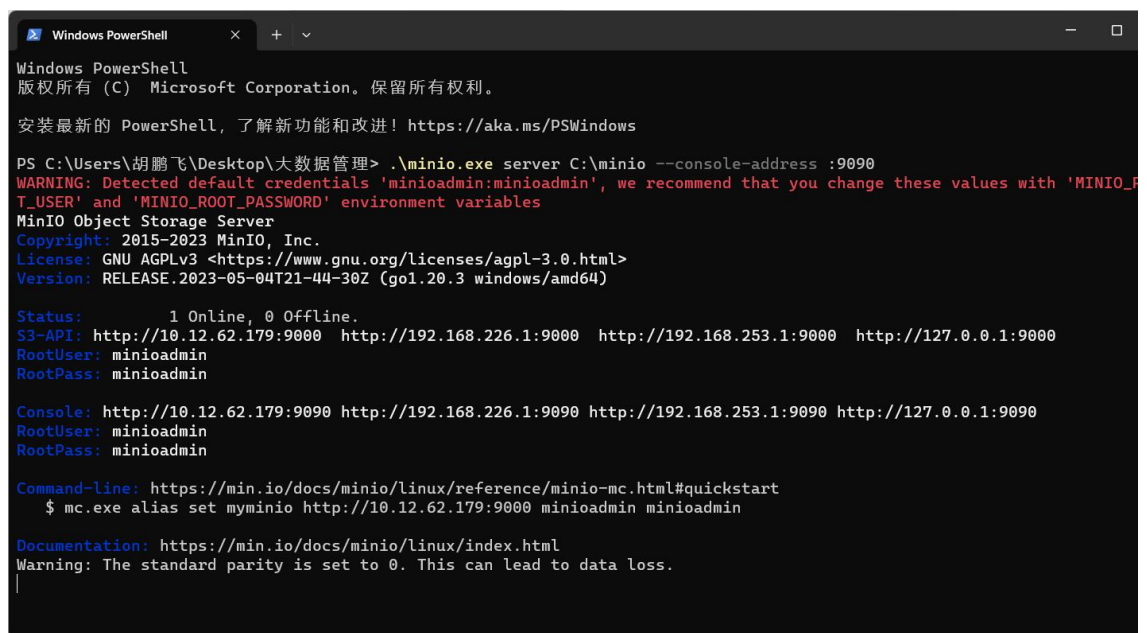
## 五、实验过程

### 5.1 对象存储技术实践

1. 下载 windows 版本 minio 服务端和 mc 客户端。
2. 进入 cmd (powershell) 运行 minio。命令行：

```
.\minio.exe server C:\minio --console-address :9090
```

命令行 1 minio 启动



```
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

安装最新的 PowerShell，了解新功能和改进！https://aka.ms/PSWindows

PS C:\Users\胡鹏飞\Desktop\大数据管理> .\minio.exe server C:\minio --console-address :9090
WARNING: Detected default credentials 'minioadmin:minioadmin', we recommend that you change these values with 'MINIO_P
T_USER' and 'MINIO_ROOT_PASSWORD' environment variables
MinIO Object Storage Server
Copyright: 2015-2023 MinIO, Inc.
License: GNU AGPLv3 <https://www.gnu.org/licenses/agpl-3.0.html>
Version: RELEASE.2023-05-04T21-44-30Z (go1.20.3 windows/amd64)

Status:          1 Online, 0 Offline.
S3-API: http://10.12.62.179:9000 http://192.168.226.1:9000 http://192.168.253.1:9000 http://127.0.0.1:9000
RootUser: minioadmin
RootPass: minioadmin

Console: http://10.12.62.179:9090 http://192.168.226.1:9090 http://192.168.253.1:9090 http://127.0.0.1:9090
RootUser: minioadmin
RootPass: minioadmin

Command-line: https://min.io/docs/minio/linux/reference/minio-mc.html#quickstart
$ mc.exe alias set myminio http://10.12.62.179:9000 minioadmin minioadmin

Documentation: https://min.io/docs/minio/linux/index.html
Warning: The standard parity is set to 0. This can lead to data loss.
```

图 1 minio 启动

3. 在浏览器打开 console 行的一个 url，并创建一个名为 test 的 bucket。

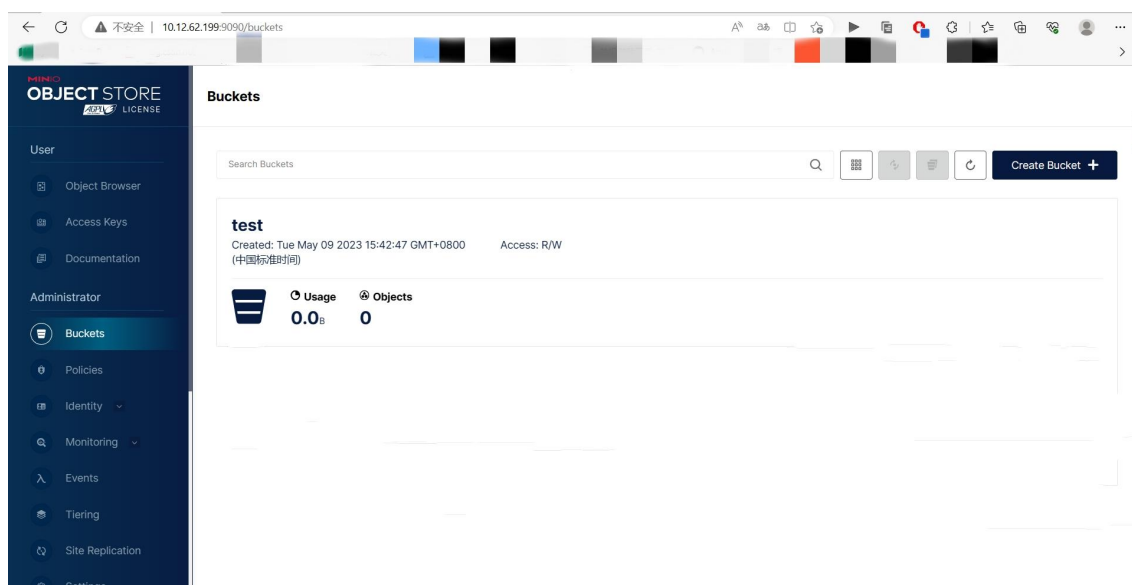


图 2 minio 页面

4. 进入 cmd (powershell) 运行 mc。

命令行:

```
.\mc config host add hust http://127.0.0.1:9000 minioadmin minioadmin --api s3v4
```

命令行 2 mc 启动

其中的 url 为图 1 minio 启动 S3-API 一行中的 url。

部分机器可能需要使用以下官方最新的命令行:

```
mc.exe alias set local http://127.0.0.1:9000 minioadmin minioadmin
```

```
mc.exe admin info local
```

命令行 3 mc 启动\_官方

启动后使用 ls 命令可以查看本地文件，使用 ls hust 命令可以查看服务器的文件。使用 mb 命令可以添加新的 bucket。效果如图所示:

```
PS C:\Users\胡鹏飞\Desktop\大数据管理> .\mc config host add hust http://127.0.0.1:9000 minioadmin minioadmin --api s3v4
Added 'hust' successfully.
PS C:\Users\胡鹏飞\Desktop\大数据管理> .\mc ls
[2023-05-09 14:30:09 CST]    0B bigdata-storage-experiment-assignment-2023/
[2023-05-09 14:34:03 CST]  25MiB mc.exe
[2023-05-09 14:33:43 CST]  96MiB minio.exe
PS C:\Users\胡鹏飞\Desktop\大数据管理> .\mc ls hust
[2023-05-09 15:42:47 CST]    0B test/
PS C:\Users\胡鹏飞\Desktop\大数据管理> .\mc mb hust/test2
Bucket created successfully 'hust/test2'.
PS C:\Users\胡鹏飞\Desktop\大数据管理> .\mc ls hust
[2023-05-09 15:42:47 CST]    0B test/
[2023-05-09 15:48:25 CST]    0B test2/
PS C:\Users\胡鹏飞\Desktop\大数据管理> |
```

图 3 mc 运行

此时，服务端网页也会更新。

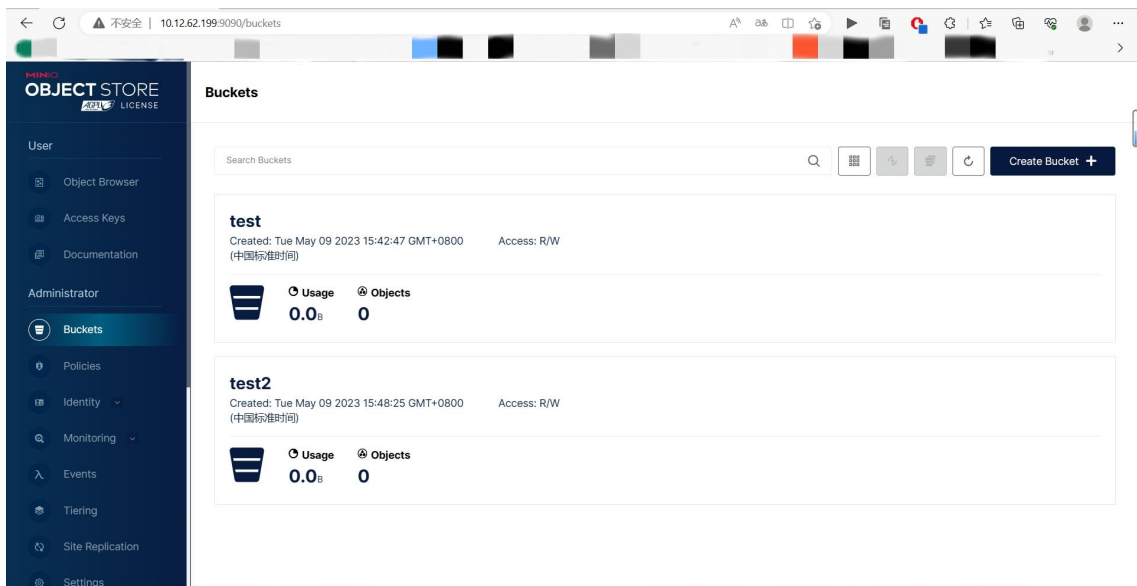


图 4 minio 页面出现新的 bucket

5. 下载 s3bench 及其脚本进行测试。

 run-s3bench.cmd	2023/5/9 17:18	Windows 命令脚本	1 KB
 s3bench.exe	2023/5/9 16:12	应用程序	10,107 KB

图 5 s3bench 及脚本

将脚本修改如下：

```
s3bench.exe ^
  -accessKey=minioadmin ^
  -accessSecret=minioadmin ^
  -bucket=test ^
  -endpoint=http://127.0.0.1:9000 ^
  -numClients=8 ^
  -numSamples=256 ^
  -objectNamePrefix=loadgen ^
  -objectSize=1024
pause
```

图 6 run-s3bench.cmd

需要注意的是-bucket 这一行要修改为已有的 bucket。

使用命令行运行脚本，得到以下结果：

```

Results Summary for Write Operation(s)
Total Transferred: 0.250 MB
Total Throughput: 1.85 MB/s
Total Duration: 0.135 s
Number of Errors: 0
-----
Write times Max: 0.015 s
Write times 99th %ile: 0.014 s
Write times 90th %ile: 0.005 s
Write times 75th %ile: 0.004 s
Write times 50th %ile: 0.004 s
Write times 25th %ile: 0.004 s
Write times Min: 0.003 s

Results Summary for Read Operation(s)
Total Transferred: 0.250 MB
Total Throughput: 7.34 MB/s
Total Duration: 0.034 s
Number of Errors: 0
-----
Read times Max: 0.003 s
Read times 99th %ile: 0.002 s
Read times 90th %ile: 0.002 s
Read times 75th %ile: 0.001 s
Read times 50th %ile: 0.001 s
Read times 25th %ile: 0.001 s
Read times Min: 0.001 s

```

图 7 测试结果

修改参数，多次进行测试，记录测试结果。

## 5.2 对象存储性能分析

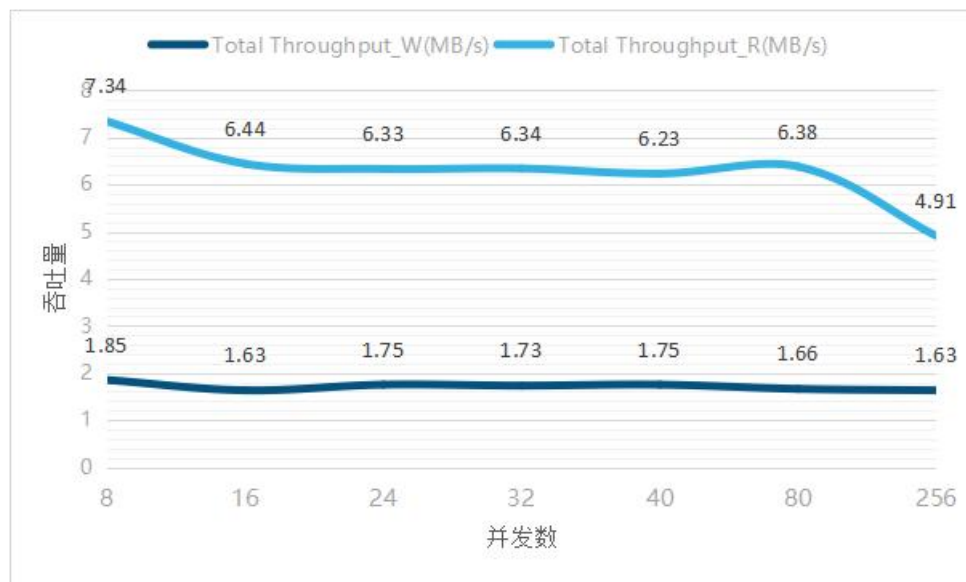
### 1. throughput 和 duration 关于 numClient 的变化

numClien	Total	Total	Total	Total
ts	Throughput_W	Duration_W	Throughput_R	Duration_R

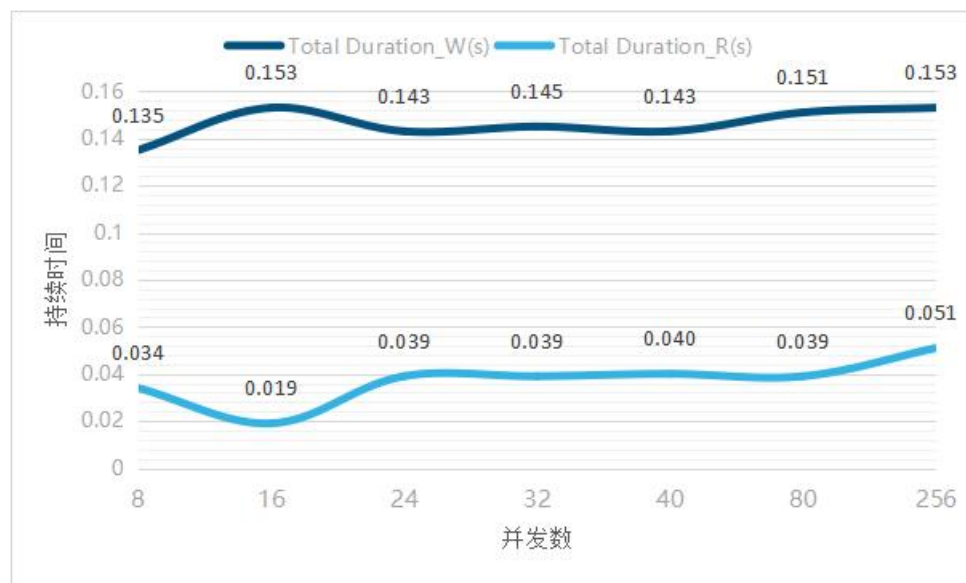


	(MB/s)	(s)	(MB/s)	(s)
8	1.85	0.135	7.34	0.034
16	1.63	0.153	6.44	0.019
24	1.75	0.143	6.33	0.039
32	1.73	0.145	6.34	0.039
40	1.75	0.143	6.23	0.040
80	1.66	0.151	6.38	0.039
256	1.63	0.153	4.91	0.051

表 1 throughput 和 duration 关于 numClient 的变化



图表 1 吞吐量随并发数的变化



图表 2 持续时间随并发数的变化

分析图表可知，在并发数增大时，吞吐量先减小后增大再减小。

在并发数增大时，持续时间在并发数较小时不断波动，而后持续增大。

由以上数据推测，当并发数较小时，随着并发数提高，系统能满足其请求，使吞吐量有一定程度的提高，当并发数较大时，出现阻塞，使吞吐量随并发数的增大而减少。

当并发数较小时，吞吐量对持续时间影响不大，但并发数过大时，导致阻塞，会使持续时间增加。

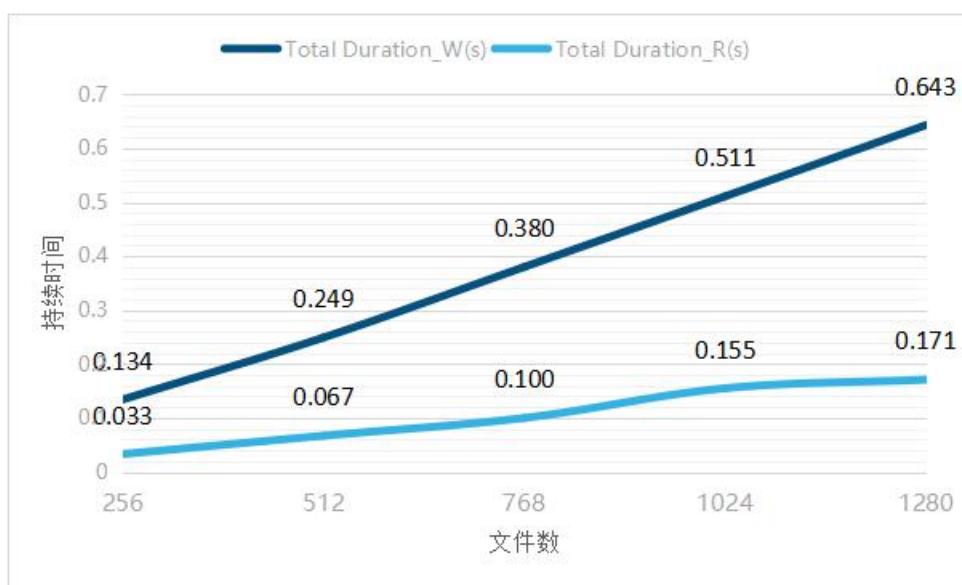
## 2. throughput 和 duration 关于 numSamples 的变化

numSamples	Total Throughput_W (MB/s)	Total Duration_W (s)	Total Throughput_R (MB/s)	Total Duration_R (s)
256	1.87	0.134	7.57	0.033
512	2.01	0.249	7.48	0.067
768	1.98	0.380	7.52	0.100
1024	1.96	0.511	6.44	0.155
1280	1.95	0.643	7.30	0.171

表 2 throughput 和 duration 关于 numSamples 的变化



图表 3 吞吐量关于文件数的变化



图表 4 持续时间关于文件数的变化

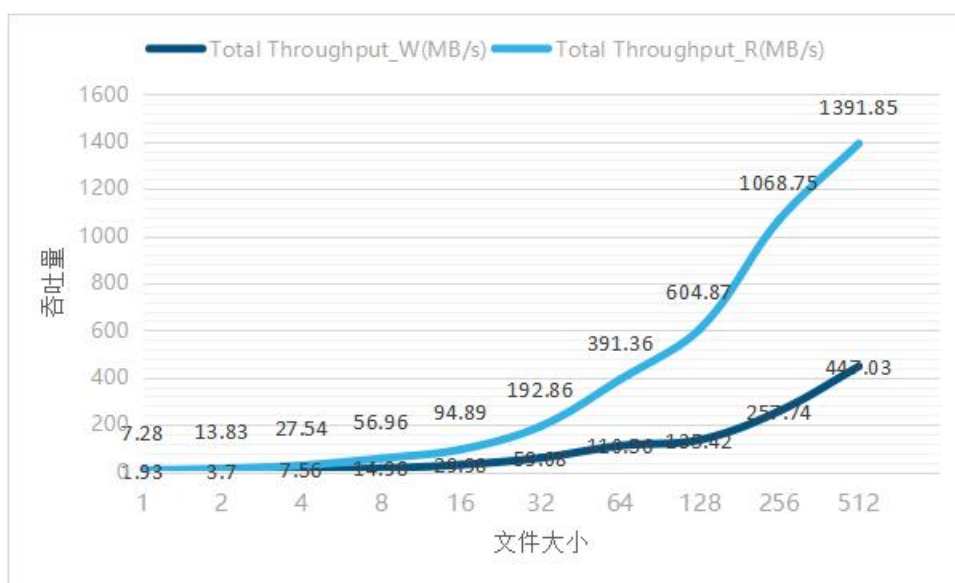
根据图表，当文件数增大时，吞吐量有所波动，但无明显变化趋势，而持续时间则稳定增加，呈线性增长。

根据以上数据推测，文件数量对吞吐量影响较小，而直接影响持续时间。持续时间与文件数呈正比。

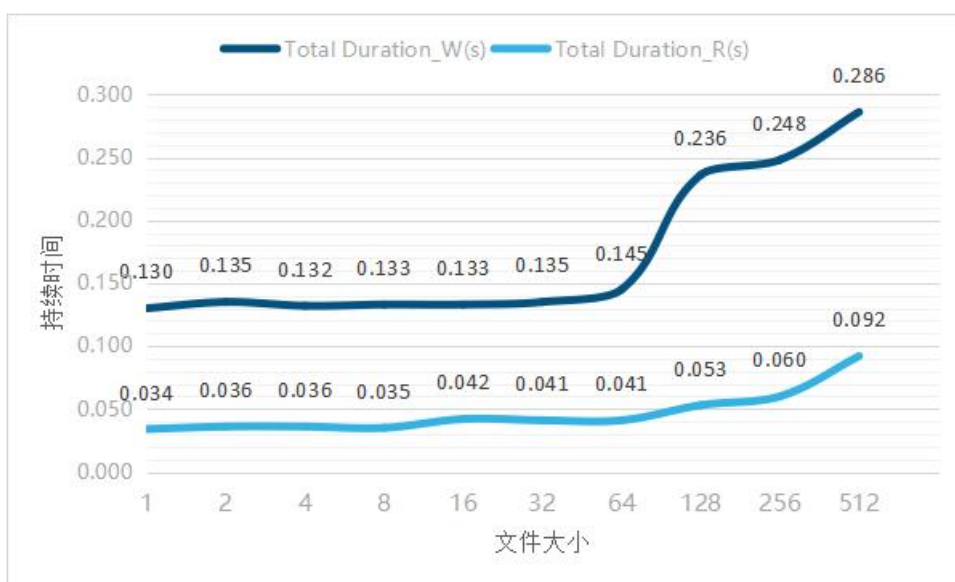
### 3. throughput 和 duration 关于 objectSize 的变化

objectSize (KB)	Total Throughput_W (MB/s)	Total Duration_W (s)	Total Throughput_R (MB/s)	Total Duration_R (s)
1	1.93	0.130	7.28	0.034
2	3.7	0.135	13.83	0.036
4	7.56	0.132	27.54	0.036
8	14.98	0.133	56.96	0.035
16	29.98	0.133	94.89	0.042
32	59.08	0.135	192.86	0.041
64	110.56	0.145	391.36	0.041
128	135.42	0.236	604.87	0.053
256	257.74	0.248	1068.75	0.060
512	447.03	0.286	1391.85	0.092

表 3 throughput 和 duration 关于 objectSize 的变化



图表 5 吞吐量关于文件大小的变化



图表 6 持续时间关于文件大小的变化

根据图表可知，随着文件大小增长，吞吐量先等比例增长，而后增长放缓，持续时间一开始无明显变化，当文件大小较大后开始增长。

根据以上数据推测，文件大小直接影响吞吐量，当文件大小较小时，吞吐量正比与文件大小，但当文件大小较大时，受硬件条件限制，吞吐量增长放缓，从而导致持续时间增加。

## 六、实验总结

在本次实验中，我完成了 minio 存储系统的部署和使用，并通过 s3bench 对其性能进行测试，得到一些列数据，并分析数据得出一些规律。

通过本次实验，我接触到了对象存储系统，对企业的存储系统有了初步认知。其次，还使用了一些自动测试工具，对测试这一步骤有了更清楚的了解。此外，还首次使用了 github 的 fork 功能，对 git 的使用更加熟练。最后，通过对测试结果的分析，得到了存储系统的一些规律，加深了对存储系统的理解。

## 参考文献

- [1] ZHENG Q, CHEN H, WANG Y 等. COSBench: A Benchmark Tool for Cloud Object Storage Services[C]//2012 IEEE Fifth International Conference on Cloud Computing. 2012: 998 - 999.
- [2] ARNOLD J. OpenStack Swift[M]. O' Reilly Media, 2014.
- [3] WEIL S A, BRANDT S A, MILLER E L 等. Ceph: A Scalable, High-performance Distributed File System[C]//Proceedings of the 7th Symposium on Operating Systems Design and Implementation. Berkeley, CA, USA: USENIX Association, 2006: 307 - 320.
- [4] Dean J, Barroso L A. Association for Computing Machinery, 2013. The Tail at Scale[J]. Commun. ACM, 2013, 56(2): 74 - 80.
- [5] Delimitrou C, Kozyrakis C. Association for Computing Machinery, 2018. Amdahl's Law for Tail Latency[J]. Commun. ACM, 2018, 61(8): 65 - 72.