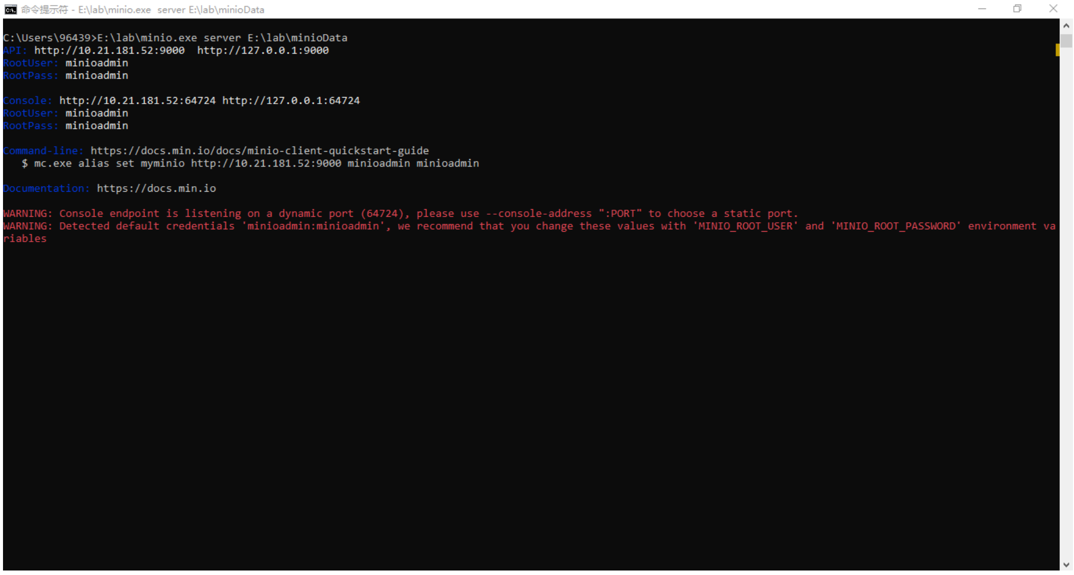
实验报告

陈端阳 M202173701

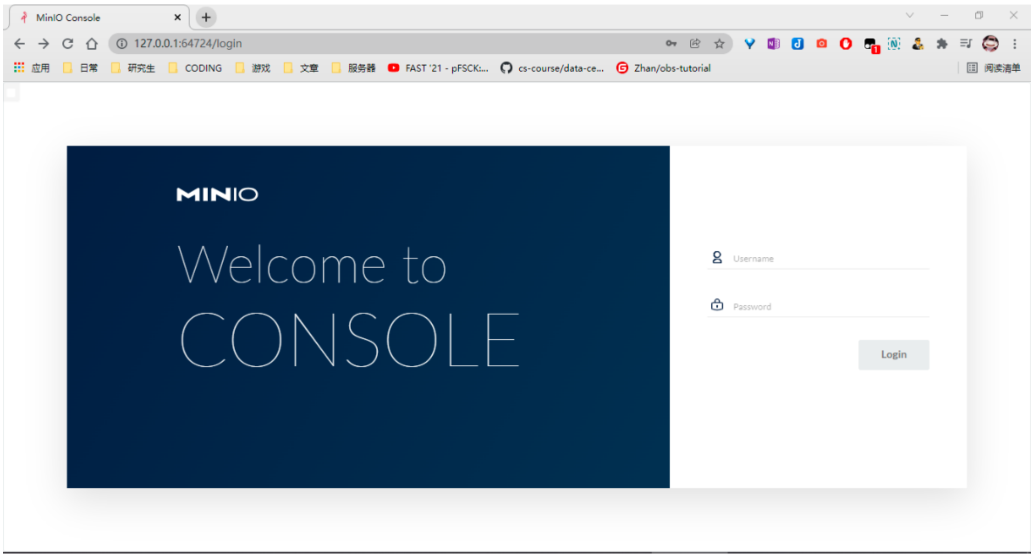
实验一 系统搭建

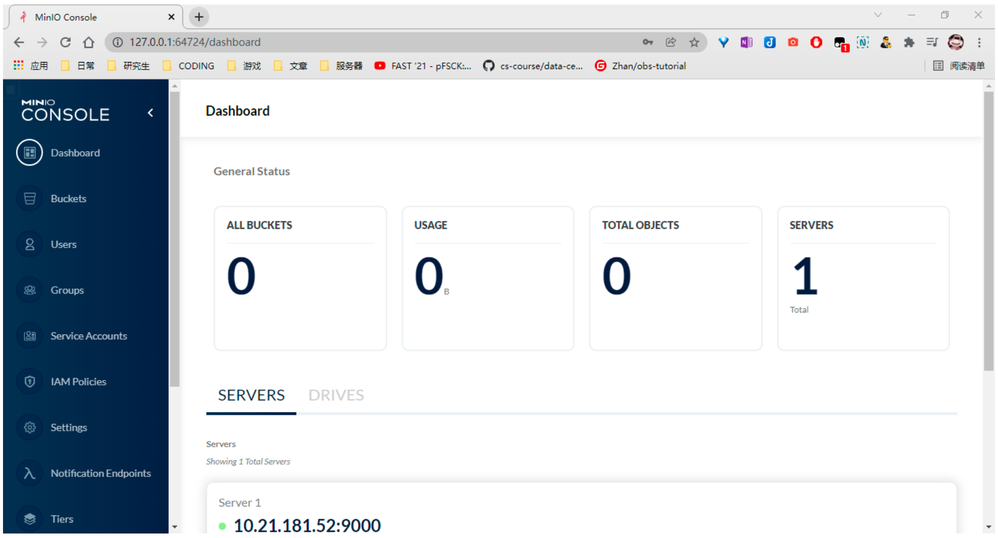
利用 cmd 启动 minio



```
C:\Users\96439>E:\lab\minio.exe server E:\lab\minioData
API: http://10.21.181.52:9000  http://127.0.0.1:9000
RootUser: minioadmin
RootPass: minioadmin

Console: http://10.21.181.52:64724 http://127.0.0.1:64724
RootUser: minioadmin
RootPass: minioadmin

Command-line: https://docs.min.io/docs/minio-client-quickstart-guide
   $ mc.exe alias set myminio http://10.21.181.52:9000 minioadmin minioadmin

Documentation: https://docs.min.io

WARNING: Console endpoint is listening on a dynamic port (64724), please use --console-address ":PORT" to choose a static port.
WARNING: Detected default credentials 'minioadmin:minioadmin', we recommend that you change these values with 'MINIO_ROOT_USER' and 'MINIO_ROOT_PASSWORD' environment va
riables
```

Chrome 登录





Dashboard

General Status

ALL BUCKETS	USAGE	TOTAL OBJECTS	SERVERS
0	0 B	0	1 Total

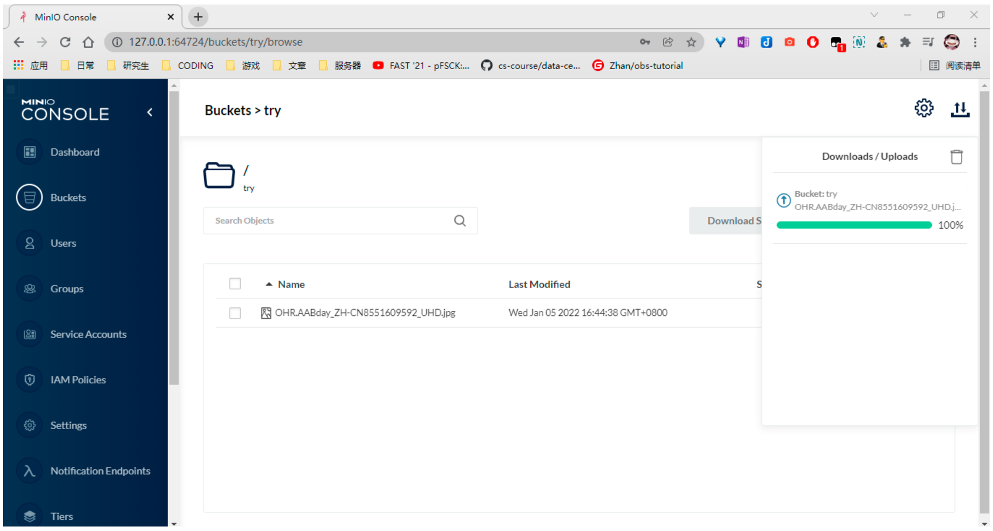
Servers

Showing 1 Total Servers

Server 1

10.21.181.52:9000

创建 bucket 并上传一张图片



Buckets > try

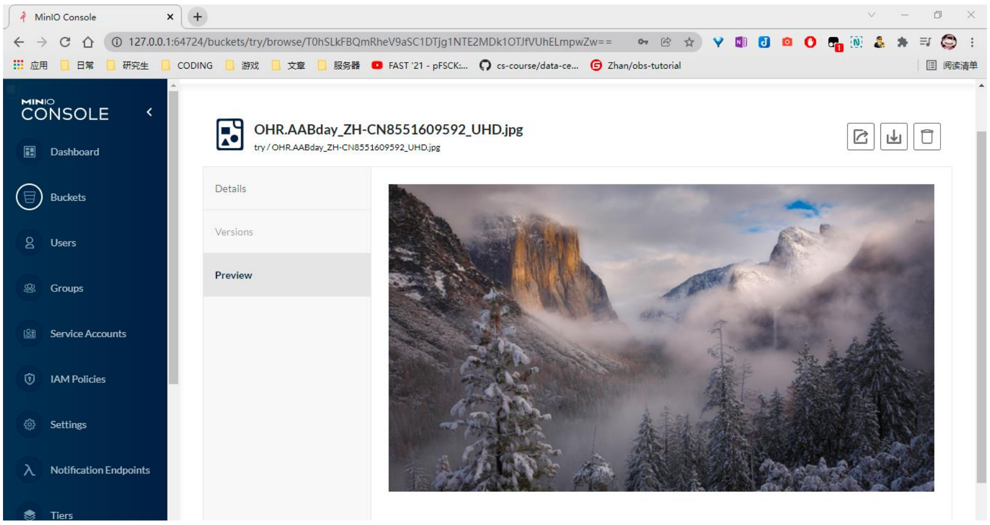
/
try

Downloads / Uploads

Bucket: try
OHR.AABday_ZH-CN8551609592_UHD.j...
100%

Name	Last Modified
OHR.AABday_ZH-CN8551609592_UHD.jpg	Wed Jan 05 2022 16:44:38 GMT+0800

在线预览刚刚上传的图片



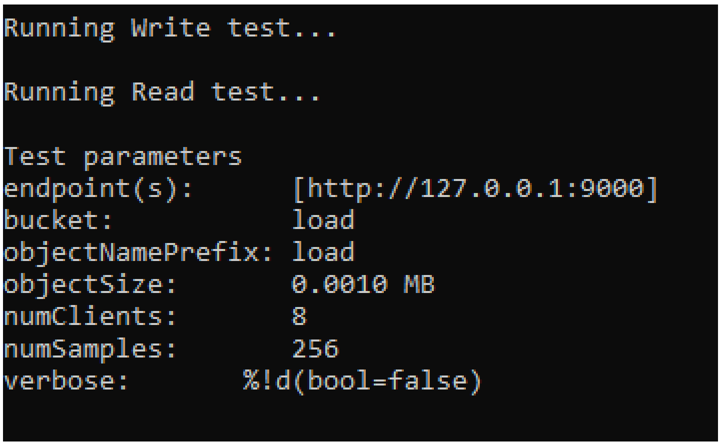
OHR.AABday_ZH-CN8551609592_UHD.jpg

try / OHR.AABday_ZH-CN8551609592_UHD.jpg

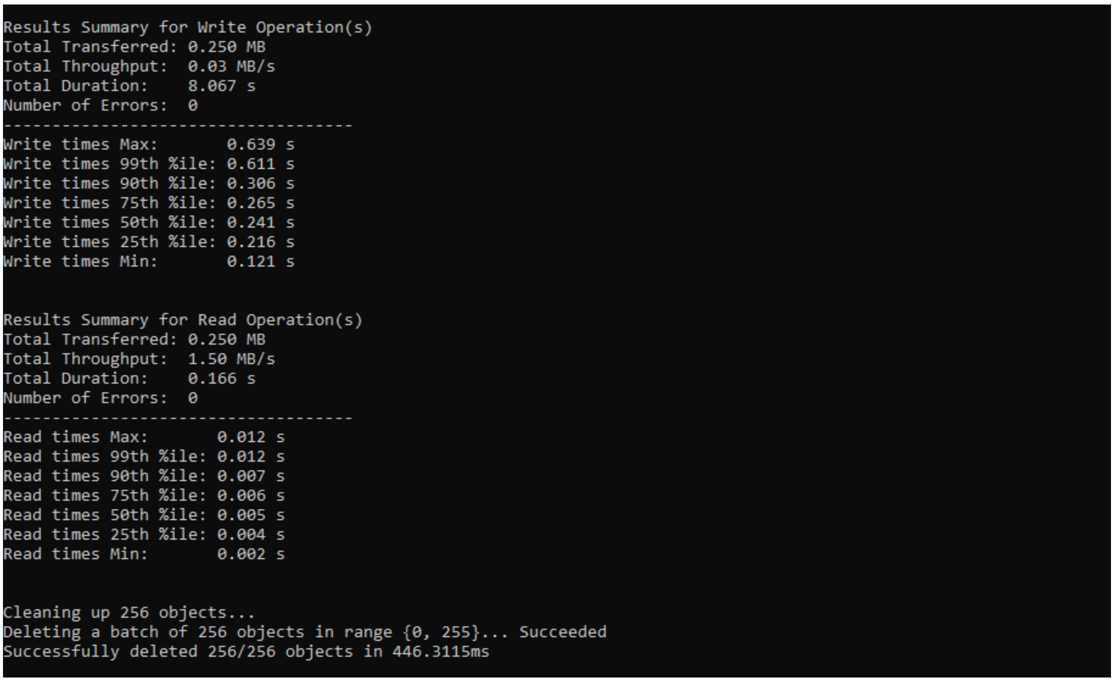
实验二 性能测试

通过 s3bench 向 minio 发送请求

256*1024KB

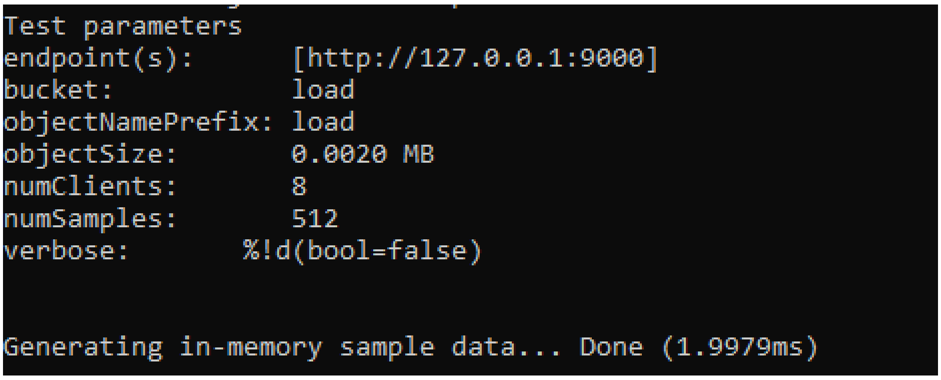


```
Running Write test...

Running Read test...

Test parameters
endpoint(s):      [http://127.0.0.1:9000]
bucket:           load
objectNamePrefix: load
objectSize:       0.0010 MB
numClients:       8
numSamples:       256
verbose:       %!d(bool=false)
```

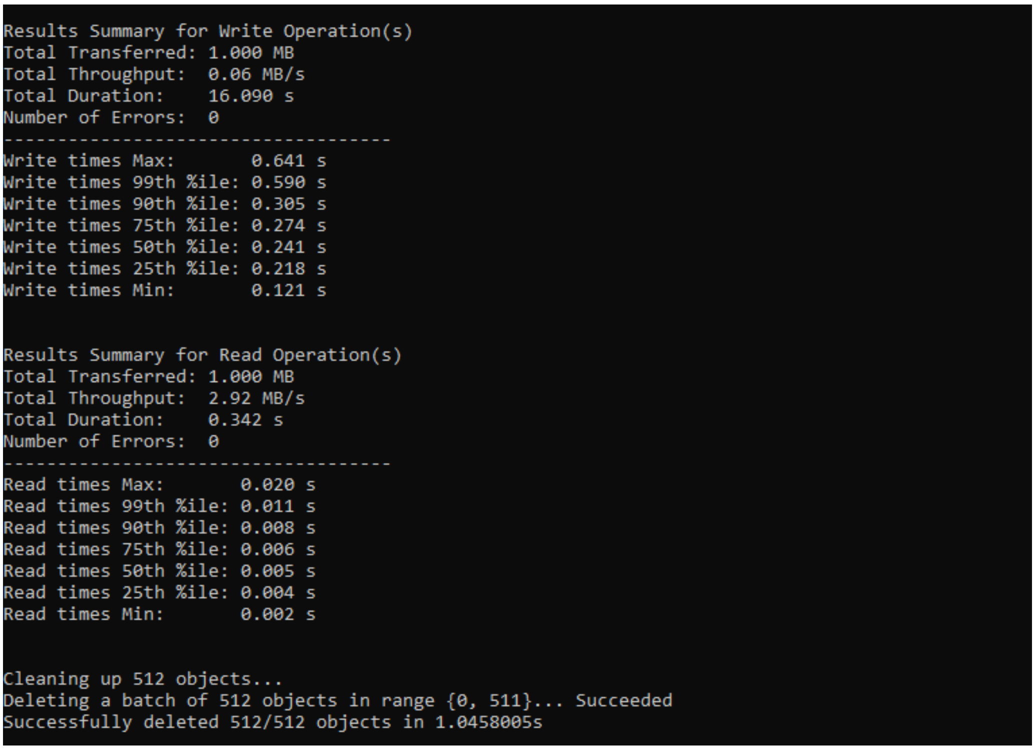


```
Results Summary for Write Operation(s)
Total Transferred: 0.250 MB
Total Throughput:  0.03 MB/s
Total Duration:    8.067 s
Number of Errors:  0
------------------------------------
Write times Max:       0.639 s
Write times 99th %ile: 0.611 s
Write times 90th %ile: 0.306 s
Write times 75th %ile: 0.265 s
Write times 50th %ile: 0.241 s
Write times 25th %ile: 0.216 s
Write times Min:       0.121 s


Results Summary for Read Operation(s)
Total Transferred: 0.250 MB
Total Throughput:  1.50 MB/s
Total Duration:    0.166 s
Number of Errors:  0
------------------------------------
Read times Max:       0.012 s
Read times 99th %ile: 0.012 s
Read times 90th %ile: 0.007 s
Read times 75th %ile: 0.006 s
Read times 50th %ile: 0.005 s
Read times 25th %ile: 0.004 s
Read times Min:       0.002 s


Cleaning up 256 objects...
Deleting a batch of 256 objects in range {0, 255}... Succeeded
Successfully deleted 256/256 objects in 446.3115ms
```

512*2048KB

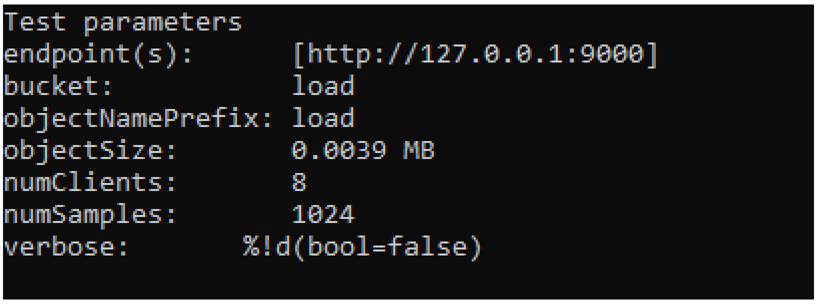


```
Test parameters
endpoint(s):      [http://127.0.0.1:9000]
bucket:           load
objectNamePrefix: load
objectSize:       0.0020 MB
numClients:       8
numSamples:       512
verbose:       %!d(bool=false)


Generating in-memory sample data... Done (1.9979ms)
```

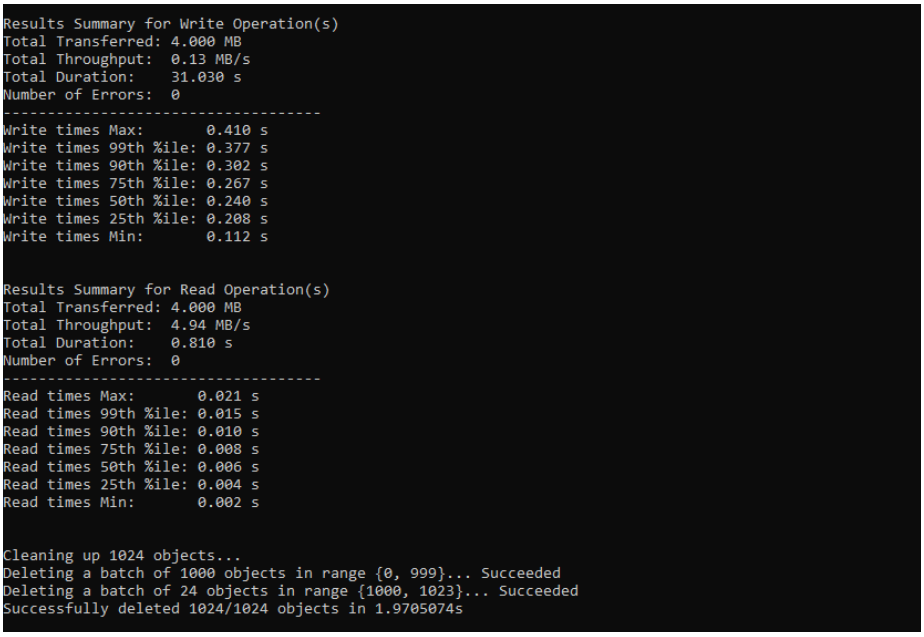


```
Results Summary for Write Operation(s)
Total Transferred: 1.000 MB
Total Throughput:  0.06 MB/s
Total Duration:    16.090 s
Number of Errors:  0
------------------------------------
Write times Max:       0.641 s
Write times 99th %ile: 0.590 s
Write times 90th %ile: 0.305 s
Write times 75th %ile: 0.274 s
Write times 50th %ile: 0.241 s
Write times 25th %ile: 0.218 s
Write times Min:       0.121 s


Results Summary for Read Operation(s)
Total Transferred: 1.000 MB
Total Throughput:  2.92 MB/s
Total Duration:    0.342 s
Number of Errors:  0
------------------------------------
Read times Max:       0.020 s
Read times 99th %ile: 0.011 s
Read times 90th %ile: 0.008 s
Read times 75th %ile: 0.006 s
Read times 50th %ile: 0.005 s
Read times 25th %ile: 0.004 s
Read times Min:       0.002 s


Cleaning up 512 objects...
Deleting a batch of 512 objects in range {0, 511}... Succeeded
Successfully deleted 512/512 objects in 1.0458005s
```

1024*4096KB



```
Test parameters
endpoint(s):      [http://127.0.0.1:9000]
bucket:           load
objectNamePrefix: load
objectSize:       0.0039 MB
numClients:       8
numSamples:       1024
verbose:       %!d(bool=false)
```

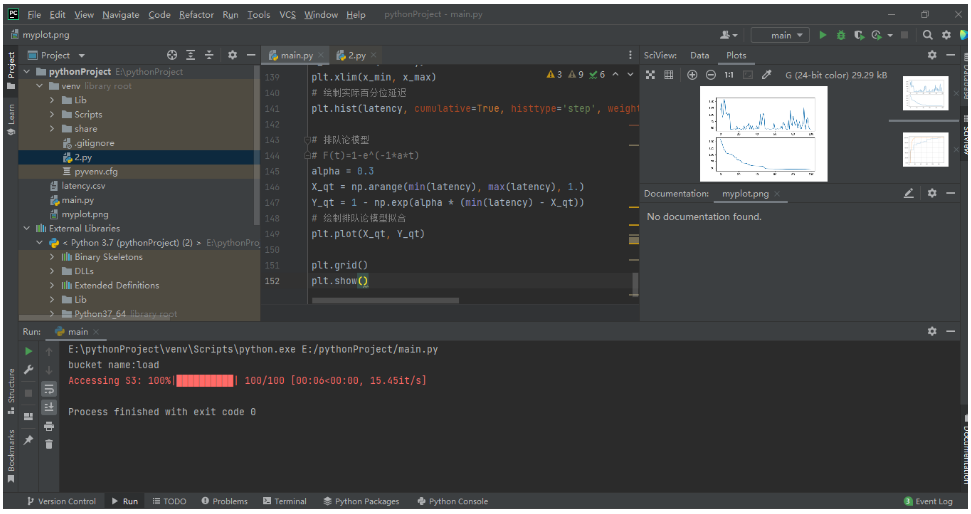


```
Results Summary for Write Operation(s)
Total Transferred: 4.000 MB
Total Throughput:  0.13 MB/s
Total Duration:    31.030 s
Number of Errors:  0
------------------------------------
Write times Max:       0.410 s
Write times 99th %ile: 0.377 s
Write times 90th %ile: 0.302 s
Write times 75th %ile: 0.267 s
Write times 50th %ile: 0.240 s
Write times 25th %ile: 0.208 s
Write times Min:       0.112 s


Results Summary for Read Operation(s)
Total Transferred: 4.000 MB
Total Throughput:  4.94 MB/s
Total Duration:    0.810 s
Number of Errors:  0
------------------------------------
Read times Max:       0.021 s
Read times 99th %ile: 0.015 s
Read times 90th %ile: 0.010 s
Read times 75th %ile: 0.008 s
Read times 50th %ile: 0.006 s
Read times 25th %ile: 0.004 s
Read times Min:       0.002 s


Cleaning up 1024 objects...
Deleting a batch of 1000 objects in range {0, 999}... Succeeded
Deleting a batch of 24 objects in range {1000, 1023}... Succeeded
Successfully deleted 1024/1024 objects in 1.9705074s
```

可以发现共同的特点，有部分的写请求用时远远超过前面的写请求

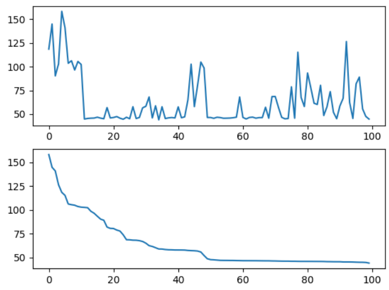
实验三 尾延迟

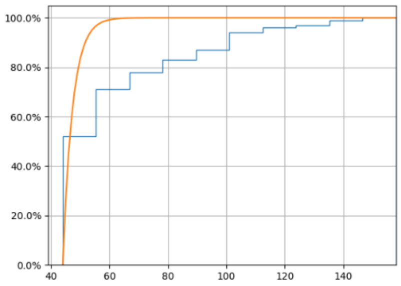
运行 obs-tutorial 中的 python 脚本，得到结果如下：



```python
plt.xlim(x_min, x_max)
# 绘制实际百分位延迟
plt.hist(latency, cumulative=True, histtype='step', weight

# 排队论模型
# F(t)=1-e^(-1*a*t)
alpha = 0.3
X_qt = np.arange(min(latency), max(latency), 1.)
Y_qt = 1 - np.exp(alpha * (min(latency) - X_qt))
# 绘制排队论模型拟合
plt.plot(X_qt, Y_qt)

plt.grid()
plt.show()
```

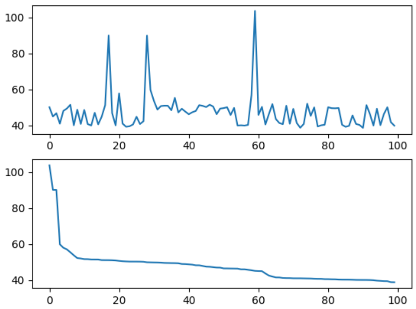
```
E:\pythonProject\venv\Scripts\python.exe E:/pythonProject/main.py
bucket name:load
Accessing S3: 100%|██████████| 100/100 [00:06<00:00, 15.45it/s]

Process finished with exit code 0
```

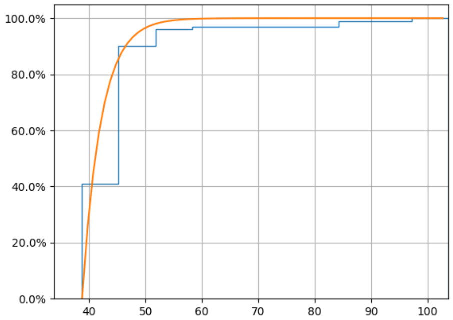




上图中存在一部分写请求的开销远超过其他写请求，就是尾延迟现象的表现。

尝试对冲





由上图可以看到 120ms 时候有 95%的数据请求发送完成，所以通过设置时间阈值为 120ms，超时后重发相同请求，得到上图结果，可以看到比之前的结果有明显改善。