

**数据中心技术课程实验报告**

院 系 计算机科学与技术

班 级 CS2106

学 号 M202173711

姓 名 翟明晗

2022年 1月 1 日

1. 系统搭建
2. 实验环境

CPU：AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz

内存：16GB

系统：Windows10

1. 下载minio，在bin目录下创建data文件夹作为数据存储的文件夹，以管理员身份打开cmd，运行run-minio.cmd程序，搭建MinIO服务器。服务器成功启动后会在cmd输出默认的用户名和密码，为hust、hust\_obs。

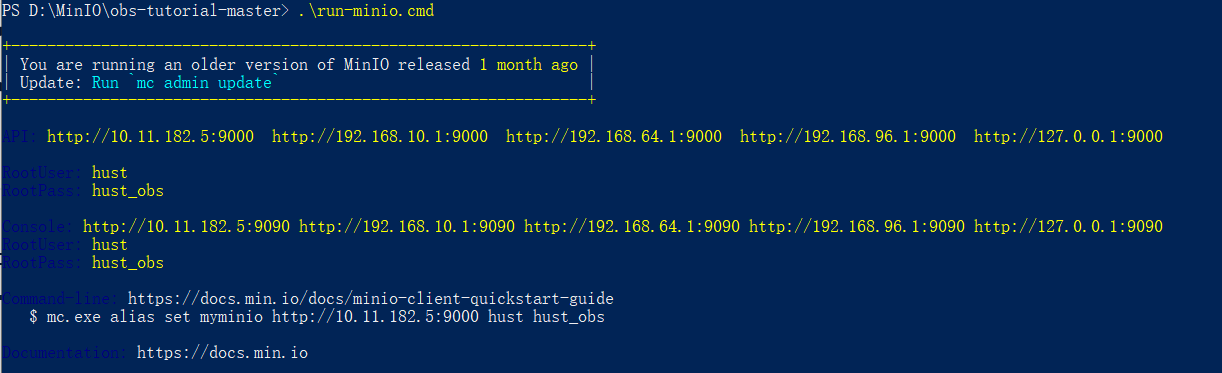


图1

1. 在浏览器中访问cmd输出的网址<http://10.11.182.5:9000>，进入minio服务器端的图形管理界面。

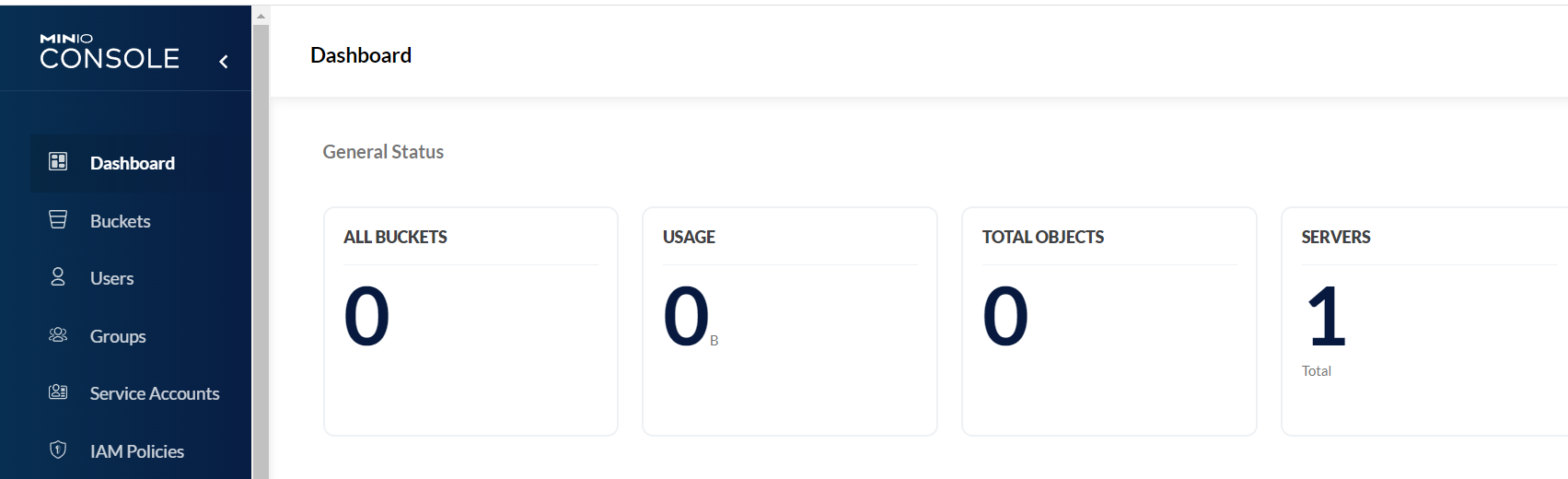


图2

1. 在buckets面板上创建一个名为“loadgen”的bucket桶，用于性能测试。

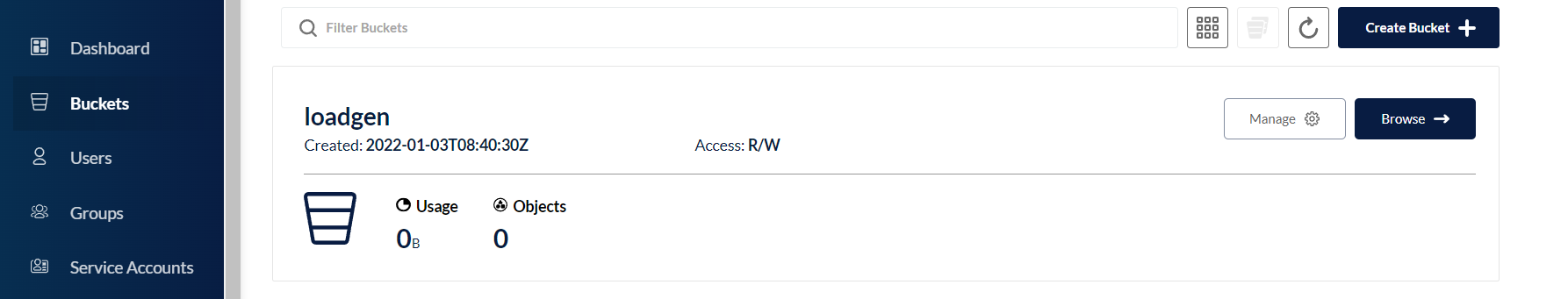
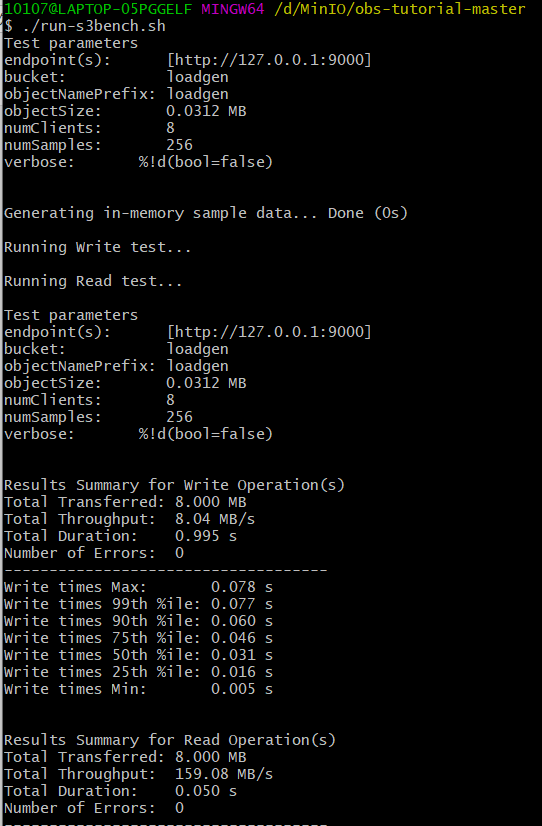


图3

1. 性能测试
2. 在git bash上执行run-s3bench.sh命令开始s3bench基准测试。



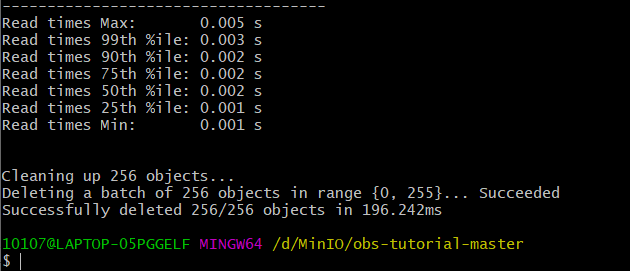


图4

由图4运行结果所示，测试文件大小为0.0312MB，共256个。其中，写操作总吞吐率为8.04MB/s。写操作总时间为0.995s，其中最大写时长为0.078s，最小写时长为0.005s，其中90%的写操作在0.06s时完成，99%的写操作在0.077s时完成。而读操作总吞吐量为159.08MB/s，读操作总时间为0.05s，其中最大读时长为0.005s，最小读时长为0.001s，其中90%的读操作在0.002s时完成，99%的读操作在0.003s时完成。

1. 尾延迟挑战

修改latency-collect.ipynb和latency-plot.Ipynb的代码(ip:port)以连接本地的MinIO服务器，首先执行latency-collect.ipynb代码获取尾延迟分布数据，并写入到latency.csv文件中，之后执行latency-plot.ipynb文件画出尾延迟分布图象如图5所示，存在一部分写请求耗时远大于其他的写请求，即为尾延迟现象。

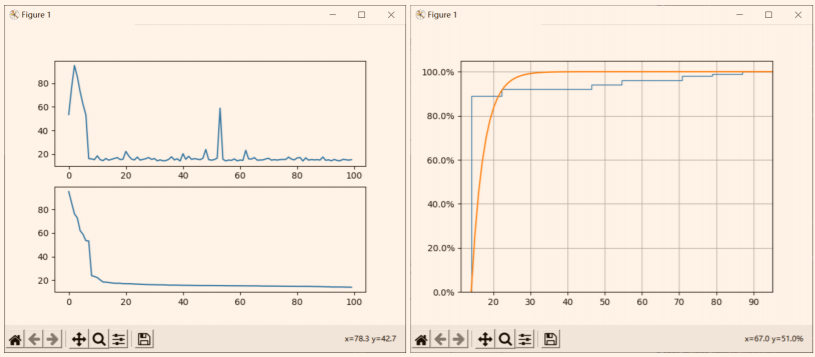


图5

尝试对冲

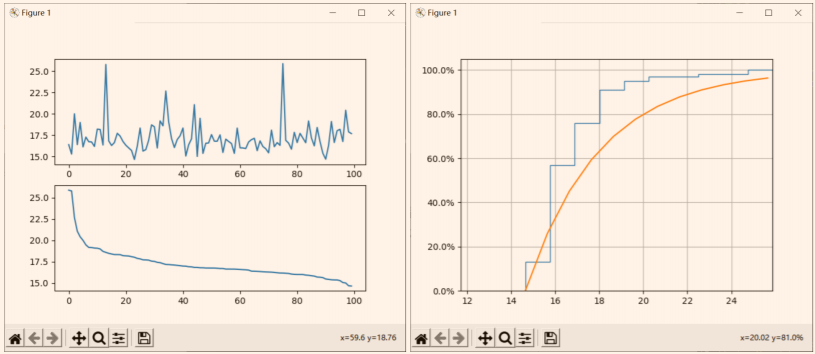


图6

由图6所示可得请求发送90%数据时时间为0.05s，所以可设置timeout为0.05s，超时重新请求，得如图所示结果，效果由明显提升。