目录

[一、 系统搭建 2](#_Toc91790150)

[实验环境 2](#_Toc91790151)

[服务器搭建：Minio 2](#_Toc91790152)

[客户端安装：Minio Client 3](#_Toc91790153)

[二、 性能观测 4](#_Toc91790154)

[安装s3bench工具 4](#_Toc91790155)

[使用s3bench测试 5](#_Toc91790156)

[三、 尾延迟挑战 8](#_Toc91790157)

[原始尾延迟 8](#_Toc91790158)

[对冲请求Hedged Request 8](#_Toc91790159)

[关联请求Tied Request 9](#_Toc91790160)

# 系统搭建

## 实验环境

操作系统：VMWare下的 Ubuntu 20.04

内核版本：linux 5.11.0

## 服务器搭建：Minio

1. 下载资料库：

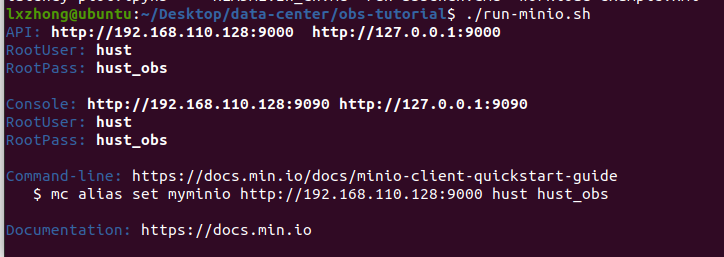
git clone https://gitee.com/shi\_zhan/obs-tutorial

1. 下载minio：（bash进入资料库目录）

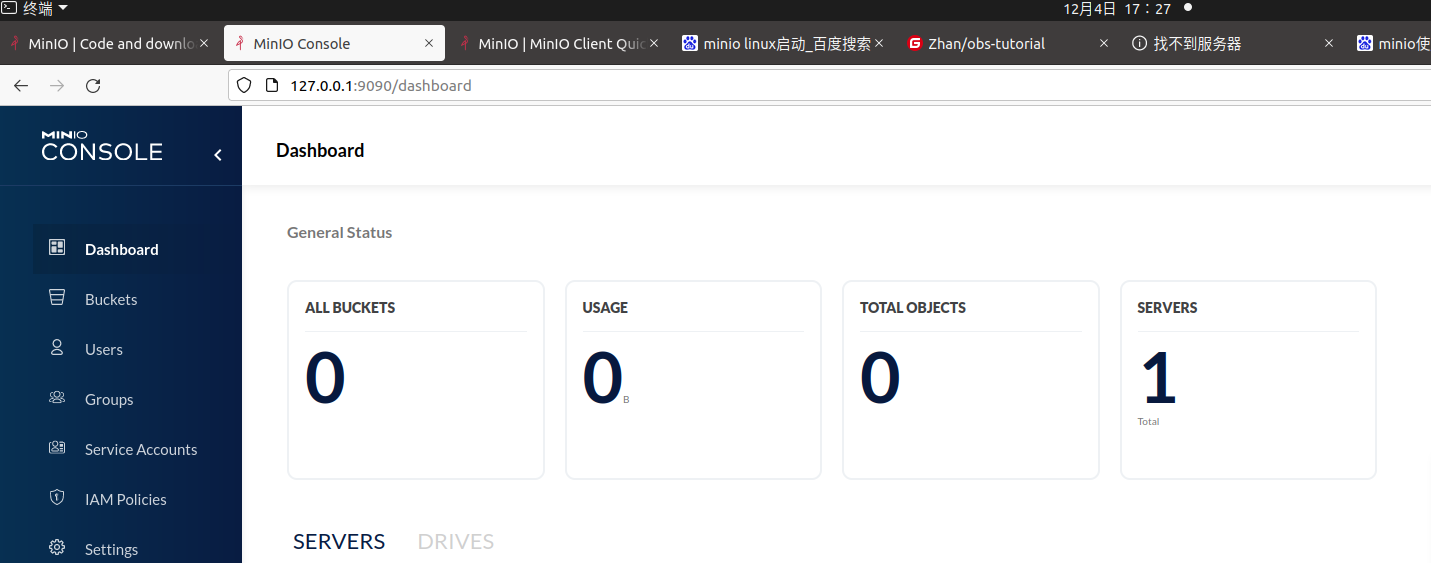
wget <https://dl.min.io/server/minio/release/linux-amd64/minio>

chmod +x minio

1. 配置帐号密码，执行run-minio.sh



1. 打开浏览器进入127.0.0.1:9000



## 客户端安装：Minio Client

1. 安装mc

wget <https://dl.min.io/client/mc/release/linux-amd64/mc>

chmod +x mc

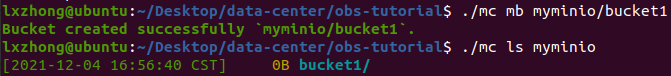
1. 输入上面的command-line，添加服务器

./mc alias set myminio http://192.168.110.128:9000 hust hust\_obs



1. 添加一个bucket：

./mc mb myminio/bucket1



# 性能观测

## 安装s3bench工具

1. 安装go环境：

sudo apt-get install golang

1. 安装s3bench

go get -u github.com/igneous-systems/s3bench

1. 修改run-s3bench.sh中的-bucket参数为自己的bucket名：

$s3bench \

-accessKey=hust \

-accessSecret=hust\_obs \

-bucket=bucket1 \

-endpoint=http://127.0.0.1:9000 \

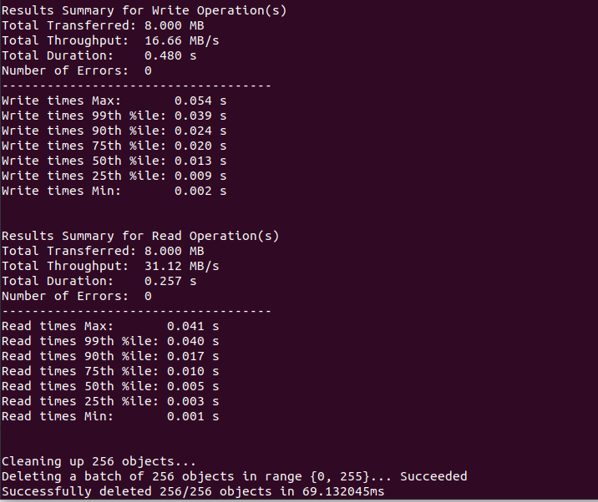
-numClients=8 \

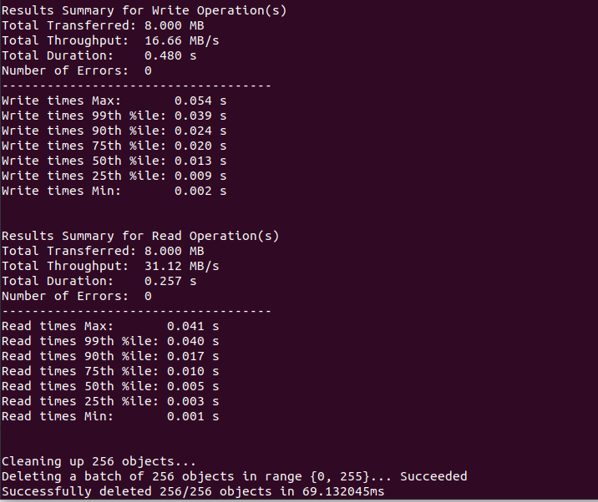
-numSamples=256 \

-objectNamePrefix=loadgen \

-objectSize=$(( 1024\*32 ))

运行结果：

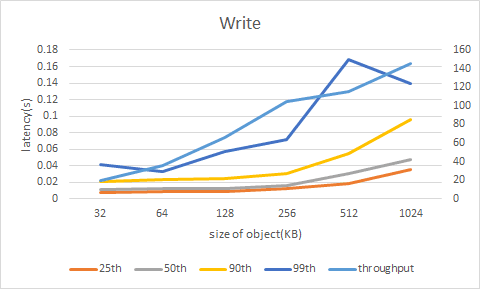


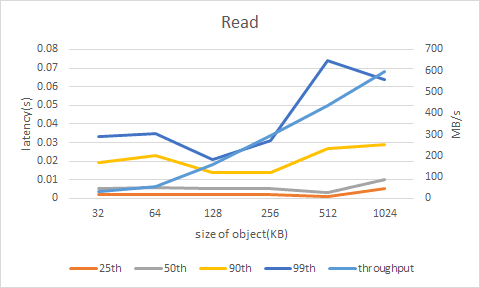


## 使用s3bench测试

1. 对象尺寸对性能的影响

选取对象尺寸分别为：32, 64, 128, 256, 512, 1024（KB），观察吞吐量和尾延迟。





观察可得：

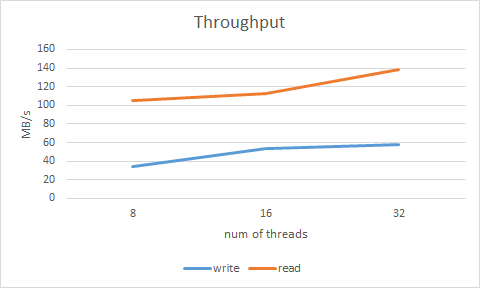
对象尺寸越大，读写吞吐量越大；

对象尺寸越大，写的中位数延迟和尾延迟越大，读操作亦有此趋势，但在尺寸很小（<128KB）时，现象不明显。

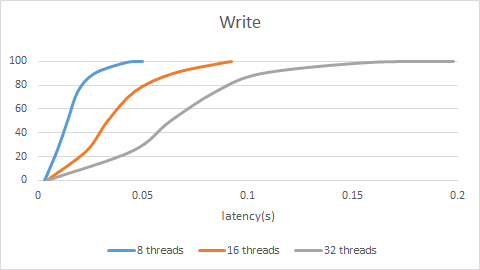
1. 并发数对性能的影响

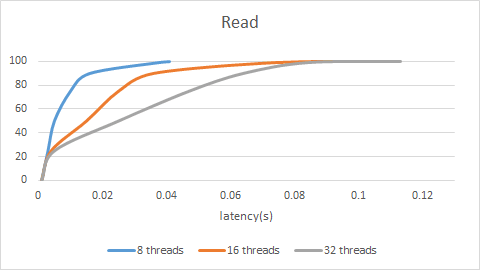
设置object size = 128 KB，分别测试线程数为8, 16, 32时的性能。

吞吐量：



读写尾延迟：



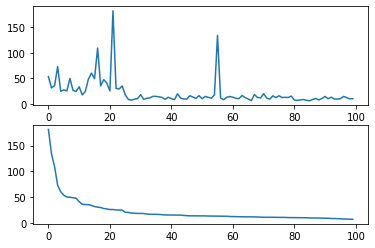


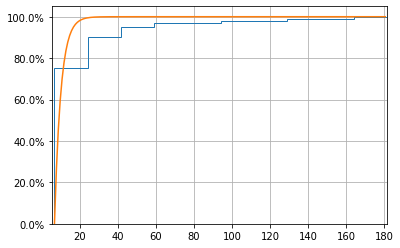
观察可得：

并发数越大，吞吐量越高，尾延迟也越大。

# 尾延迟挑战

## 原始尾延迟

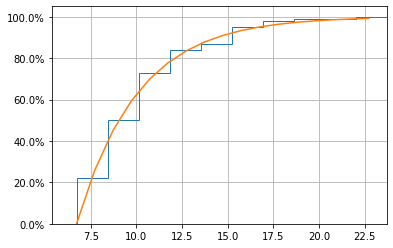




## 对冲请求Hedged Request

原理：向多个副本发出相同的请求，然后使用首先响应的副本的结果。

测试两次，假设为同时对两个服务器的两次请求，取每一对延迟中的较低者作为结果。

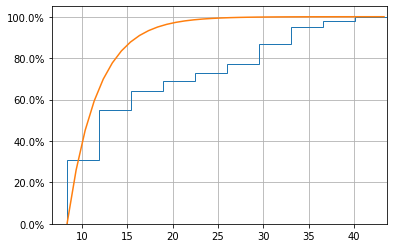


尾延迟明显降低，99th延迟也达到22ms以内。

## 关联请求Tied Request

原理：对冲请求技术也有一个漏洞，多个服务器可能不必要地执行相同的请求。在发出被对冲的请求之前，可以等待P95的预期延迟，但只有一小部分的请求可以受益。

95th延迟大概为20ms左右，对延迟超过20ms的请求，重新请求一次。



尾延迟相对原始尾延迟结果有较大改善，99th延迟降到40ms以内。没有2中对冲方案优秀，但是此方案的优点在于节省了额外开销，避免两个服务器同时处理一个相同请求。