

**2017** 级

《物联网数据存储与管理》课程

**实 验 报 告**

**姓 名 湛培向**

**学 号 U201714505**

**班 号 物联网1701班**

**日 期 2020.06.01**

**目 录**

[一、实验目的 1](#_Toc509412095)

[二、实验背景 1](#_Toc509412096)

[三、实验环境 1](#_Toc509412097)

[四、实验内容 1](#_Toc509412098)

[4.1 对象存储技术实践 1](#_Toc509412099)

[4.2 对象存储性能分析 1](#_Toc509412100)

[五、实验过程 1](#_Toc509412101)

[六、实验总结 8](#_Toc509412102)

[参考文献 8](#_Toc509412103)

# 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，架设实际应用，示范主要功能。

# 二、实验背景

随着万维网的持续进化和物联网的持续发展，当今数据存储存在着巨大的挑战。一方面数据庞大，规模持续扩张；另一方面，数据的内容结构非常丰富。传统模式的存储技术效率低且管理复杂，而新产生的对象存储技术具有性能高、管理简单的特点。它具有存储“数据”和“属性”的特点，使它更可以满足未来的扩展操作。

Mock-S3，是用Python重写fake-S3实现的，沙盒环境中测试非常有用，无需实际调用Amazon，其目标是最小化运行时依赖关系，并更像是一个开发工具来测试代码中的S3调用。

S3cmd是一款免费的命令行工具和客户端，用于在Amazon S3和其他使用S3协议的云存储服务提供商中上传，检索和管理数据。

# 三、实验环境

表1 实验环境说明

|  |  |
| --- | --- |
| 操作系统 | Ubuntu 16.04 LTS |
| 系统内核版本 | Linux version 4.15.0 |
| CPU | Intel® Core™ i7-6700HQ CPU @ 2.60GHz × 2 |
| Python版本 | 2.7.12 |
| Golang版本 | 1.14.3 |

# 四、实验内容

1.熟悉Git与linux环境，安装Python，Go以及Java基础环境。

2.实践对象存储，配置服务端与客户端，并进行基础的操作尝试。

3.对配置好的对象存储系统进行测试，并分析性能。

## 4.1 对象存储技术实践

1.在linux下配置Python，Go等运行环境。

2.安装mock-s3作为服务器端，安装s3cmd作为客户端。

3.使用例如新建bucket、上传删除文件等指令简单测试系统。

## 4.2 对象存储性能分析

1.安装测试用工具s3-benchmark。

2.改变各种不同的参数，记录每次的测试结果。

3.根据测试结果，分析不同数据的影响。

# 五、实验过程

（编号说明实验过程及观测效果、数据、图表）

1.首先配置实验环境，包括Python和Go，配置完成好的环境如下图所示。

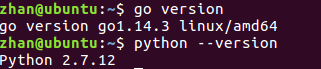


图1 Python及Go所用版本

2.安装搭建mock-s3服务端，直接从github上获取项目源码，根据教程安装启动即可。

3.安装配置s3cmd，直接输入pip install s3cmd运行。待安装完成后，配置access\_key,secret\_key,host\_base以及host\_bucket等信息。之后通过s3cmd，创建bucket并查看。

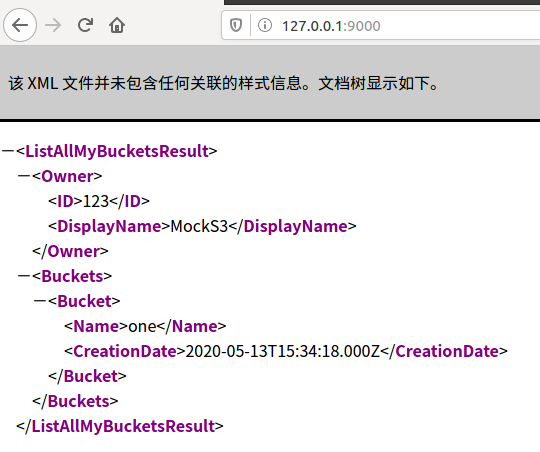


图2 创建的一个名为one的bucket



图3 使用s3cmd命令查看所有bucket

随后向新建的bucket上传一个测试文件。



图4 向bucket上传测试文件

之后在浏览器中查看，发现文件已经上传成功。



图5 在浏览器中查看文件

再使用命令删除这个文件。



图6 使用s3cmd删除文件

至此，对于服务端与客户端的安装和基本操作测试完成。下面使用s3-benchmark进行性能测试和分析。

使用命令go get -u github.com/chinglinwen/s3-benchmark安装测试工具，安装完毕后，使用./s3-benchmark -a 123 -s 123 -u http://127.0.0.1:9000 -b one -d 3 -t 1 -z 1k先进行测试，看安装是否成功。

测试时服务端在不断返回数据，而测试工具也给出了最后的结果，由此可见测试成功。

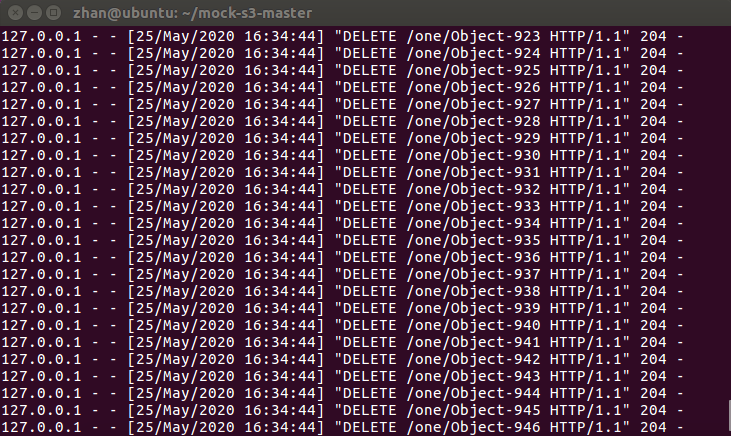


图7 服务端在测试时返回数据

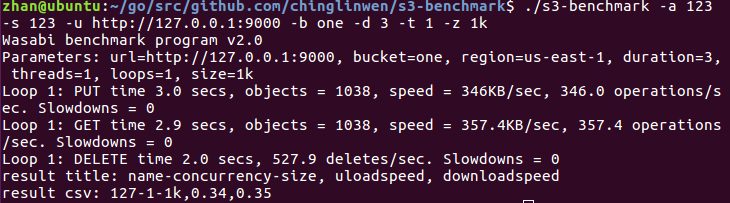


图8 s3-benchmark的命令界面

之后每次改变一个参数，分析各参数对于性能的影响。

在参数t为1,z为1k的情况下，改变d的数据，得到的数据如下所示。

表2 只改变参数d时各项数据的测试表



根据数据做出统计图

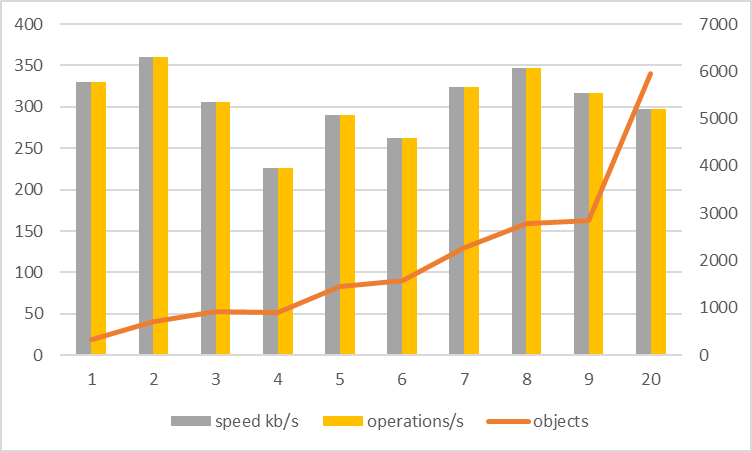


图9 PUT操作时各项数据随d的变化图

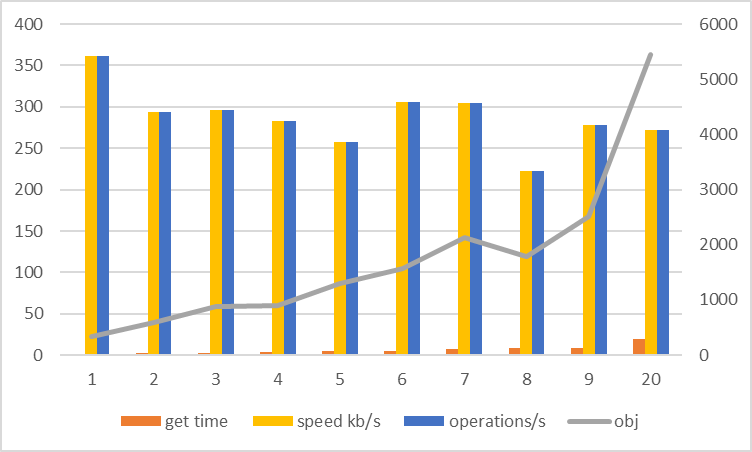


图10 GET操作时各项数据随d的变化图

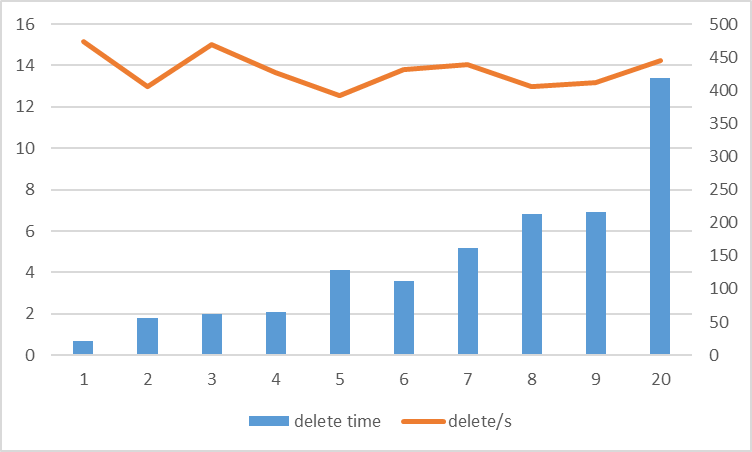


图11 delete操作时各项数据随d的变化图

从以上的数据图表中可以看出，随着操作时间的增长，所能进行的总的操作数是在不断增加的，但是每秒钟所能进行的操作数目大致相同。总体来看，put与get操作的速度大致相同，而进行删除操作时的速度快于其他操作。

之后令d=3,z为1k的情况下，改变参数t的值，即线程数。得到的数据统计表如下所示

表3 只改变参数t时各项数据的测试表



根据统计表，做出统计图如下所示。

图12 PUT操作时各项数据随t的变化图

图13 GET操作时各项数据随t的变化图

图14 delete操作时各项数据随t的变化图

从以上的数据分析，可以得出，随着线程数的增加，进行各项操作时的速度都有所增加，但是在t大于4之后，这种增长的趋势开始变得不明显，甚至有下降的趋势，因此不是线程数越大越好。

之后令d=3,t为1的情况下，改变参数z的值，即每个测试文件的大小。得到的数据统计表如下所示。

表4 只改变参数z时各项数据的测试表



根据统计表，做出统计图如下所示。

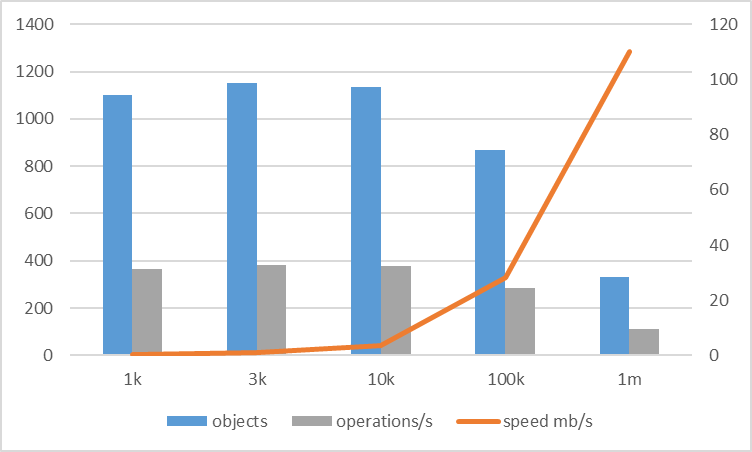


图15 PUT操作时各项数据随z的变化图

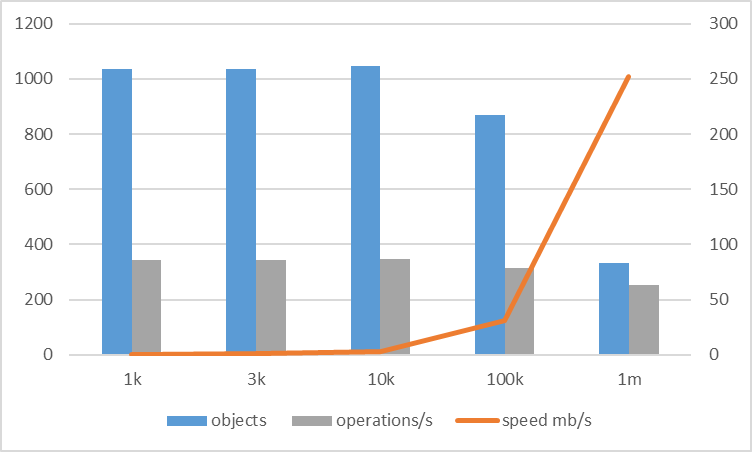


图16 GET操作时各项数据随z的变化图

图17 delete操作时各项数据随z的变化图

从以上的统计图表中分析发现，随着对象大小的增加，一定时间内能够操作的对象数目明显下降，但是速度显著提升。get与delete操作所需时间都明显下降。

# 六、实验总结

通过这次的实验课程，让我学习到了对象存储技术的相关知识，并动手实践搭建了对象存储的服务端和客户端，并通过s3-benchmark评测工具，通过改变参数的方式，对存储中的各项数据进行了测试和分析。

在这次实验的过程中，也遇到了一些问题，主要是在环境配置方面的，通过查阅网上的资料，最终还是得到了解决，并且在这个过程中学习到了很多，这些收获也会在今后的学习中对我有很大的帮助。

# 参考文献

[1] ARNOLD J. OpenStack Swift[M]. O’Reilly Media, 2014.

[2] 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.

[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.