



2015 级

《物联网数据存储与管理》课程

实 验 报 告

姓 名 陈艺欣

学 号 U201514888

班 号 物联网 1501 班

日 期 2018.04.30

目 录

一、实验目的.....	1
二、实验背景.....	1
对象存储系统（Object-Based Storage System）是综合了 NAS 和 SAN 的优点，同时具有 SAN 的高速直接访问和 NAS 的数据共享等优势，提供了高可靠性、跨平台性以及安全的数据共享的存储体系结构。	1
三、实验环境.....	1
1.1 Git 和 Github	1
1.2 Python.....	2
1.3 Java.....	2
四、实验内容.....	2
五、实验过程.....	2
2 Minio.....	2
2.1 Server.....	2
2.2 Client	3
2.3 cosbench	4
3 s3proxy.....	9
3.1 Server.....	9
3.2 Client（aws-shell）	9
3.3 cosbench	10
六、实验总结.....	12
参考文献.....	13

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

对象存储系统（Object-Based Storage System）是综合了 NAS 和 SAN 的优点，同时具有 SAN 的高速直接访问和 NAS 的数据共享等优势，提供了高可靠性、跨平台性以及安全的数据共享的存储体系结构。

三、实验环境

1.1 Git 和 Github

```
chen0@DESKTOP-DF5CIP7 MINGW64 ~  
$ git --version  
git version 2.16.2.windows.1  
  
chen0@DESKTOP-DF5CIP7 MINGW64 ~  
$ |
```

图 1 Git 版本

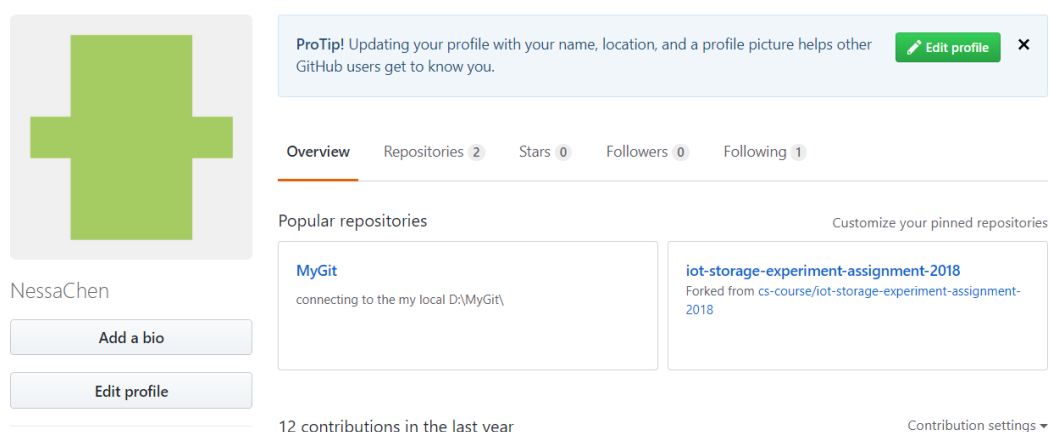
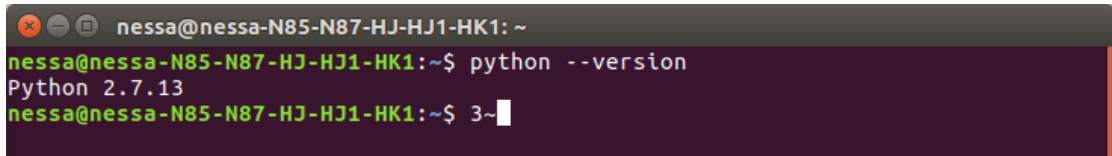


图 2 Github 账号

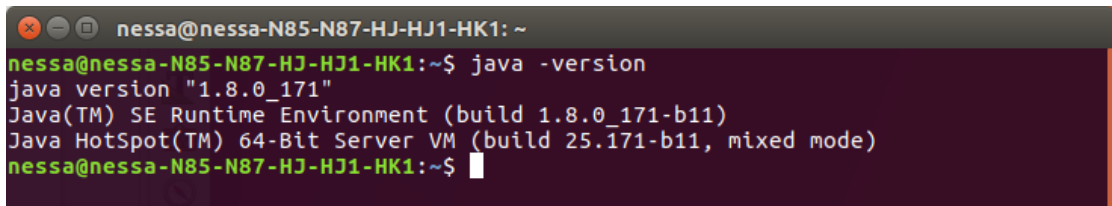
1.2 Python



```
nessa@nessa-N85-N87-HJ-HJ1-HK1: ~  
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$ python --version  
Python 2.7.13  
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$ 3~
```

图 3 Python 版本

1.3 Java



```
nessa@nessa-N85-N87-HJ-HJ1-HK1: ~  
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$ java -version  
java version "1.8.0_171"  
Java(TM) SE Runtime Environment (build 1.8.0_171-b11)  
Java HotSpot(TM) 64-Bit Server VM (build 25.171-b11, mixed mode)  
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$
```

图 4 Java 版本

四、实验内容

Server	Minio Server	s3proxy
Client	Minio Client	aws-shell
Benchmark	cosbench	cosbench

五、实验过程

1 Minio

Server

从 Minion 官网下载，以/data 文件夹作为服务器位置，通过./minio 命令运行。

```
nessa@nessa-N85-N87-HJ-HJ1-HK1: ~/下载
nessa@nessa-N85-N87-HJ-HJ1-HK1:~/下载$ ./minio server ~/StorageLab/data
Drive Capacity: 3.6 GiB Free, 13 GiB Total

Endpoint: http://127.0.0.1:9000
AccessKey: XLKXKD38YSZX9LHTN2QJ
SecretKey: 10pWfTIscPSZC30ycVmFfq5mQ1gEX0iqqWDRTopZ

Browser Access:
http://127.0.0.1:9000

Command-line Access: https://docs.minio.io/docs/minio-client-quickstart-guide
$ mc config host add myminio http://127.0.0.1:9000 XLKXKD38YSZX9LHTN2QJ 10pWf
TIscPSZC30ycVmFfq5mQ1gEX0iqqWDRTopZ

Object API (Amazon S3 compatible):
Go: https://docs.minio.io/docs/golang-client-quickstart-guide
Java: https://docs.minio.io/docs/java-client-quickstart-guide
Python: https://docs.minio.io/docs/python-client-quickstart-guide
JavaScript: https://docs.minio.io/docs/javascript-client-quickstart-guide
.NET: https://docs.minio.io/docs/dotnet-client-quickstart-guide
```

图 5 Minio 运行终端输出

运行成功后点击终端显示的地址，打开 Minio Browser

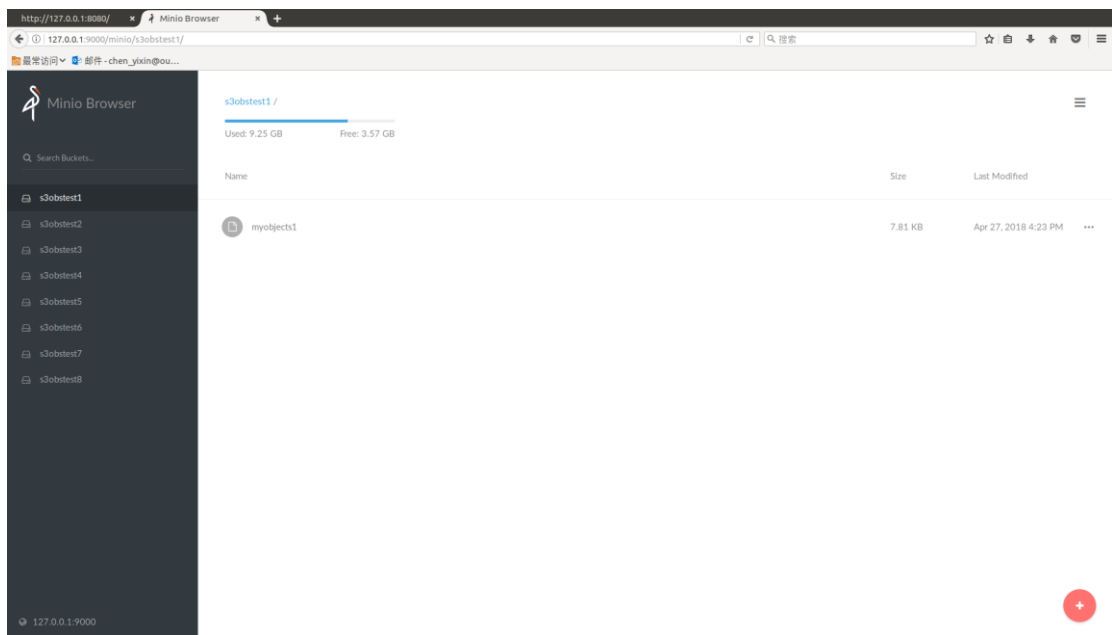


图 6 Minio Browser

Client

从官网下载 minio client，通过./mc 命令运行。先添加 minio host，然后通过 ls 命令访问 host。

```
nessa@nessa-N85-N87-HJ-HJ1-HK1:~/StorageLab$ ./mc ls minio
[2018-04-27 16:23:57 CST] 0B s3obtest1/
[2018-04-27 16:23:57 CST] 0B s3obtest2/
[2018-04-27 16:23:57 CST] 0B s3obtest3/
[2018-04-27 16:23:57 CST] 0B s3obtest4/
[2018-04-27 16:23:57 CST] 0B s3obtest5/
[2018-04-27 16:23:57 CST] 0B s3obtest6/
[2018-04-27 16:23:57 CST] 0B s3obtest7/
nessa@nessa-N85-N87-HJ-HJ1-HK1:~/StorageLab$
```

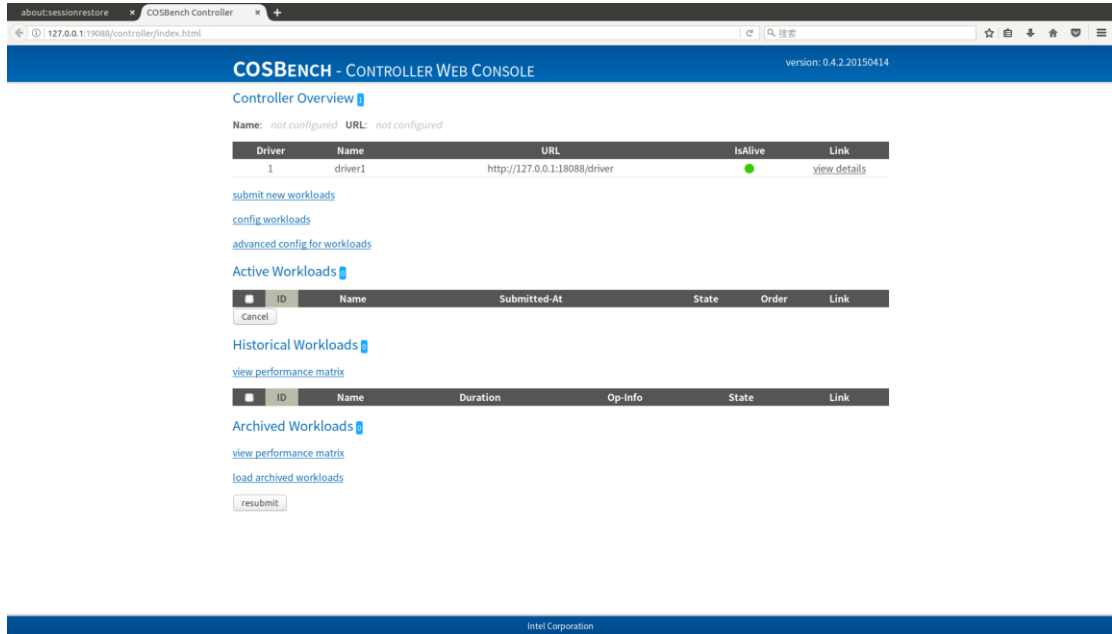



图 9 Cosbench Controller Web Console

编辑 workloads 文件。

```
<?xml version="1.0" encoding="UTF-8" ?>

<workload name="sample" description="sample benchmark">

  <storage type="s3"
  config="accesskey=V9QRTFZZW4Y0I8LESKSZ;secretkey=PwWkyupV4EoLviR55A6Dvo9B8S
  MSfq41YMKupd;proxyhost=9000;endpoint=http://127.0.0.1:9000" />

  <workflow>

    <workstage name="init">
      <work type="init" workers="1" config="cprefix=s3test;containers=r(1,8)" />
    </workstage>

    <workstage name="prepare">
      <work type="prepare" workers="8"
      config="cprefix=s3test;containers=r(1,1);objects=r(1,8);sizes=c(8)KB" />
      <work type="prepare" workers="8"
      config="cprefix=s3test;containers=r(2,2);objects=r(1,8);sizes=c(16)KB" />
      <work type="prepare" workers="8"
      config="cprefix=s3test;containers=r(3,3);objects=r(1,8);sizes=c(32)KB" />
      <work type="prepare" workers="8"
      config="cprefix=s3test;containers=r(4,4);objects=r(1,8);sizes=c(64)KB" />
      <work type="prepare" workers="8"
      config="cprefix=s3test;containers=r(5,5);objects=r(1,8);sizes=c(128)KB" />
    </workstage>
  </workflow>
</workload>
```

```

    <work type="prepare" workers="8"
config="cprefix=s3test;containers=r(6,6);objects=r(1,8);sizes=c(256)KB" />
    <work type="prepare" workers="8"
config="cprefix=s3test;containers=r(7,7);objects=r(1,8);sizes=c(512)KB" />
    <work type="prepare" workers="8"
config="cprefix=s3test;containers=r(8,8);objects=r(1,8);sizes=c(1)MB" />
</workstage>

<workstage name="8kb">
    <work name="8kb" workers="8" runtime="30">
        <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(1);objects=u(1,8)" />
        <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(8)KB" />
    </work>
</workstage>

<workstage name="16kb">
    <work name="16kb" workers="8" runtime="30">
        <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(2);objects=u(1,8)" />
        <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(16)KB" />
    </work>
</workstage>

<workstage name="32kb">
    <work name="32kb" workers="4" runtime="30">
        <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(3);objects=u(1,8)" />
        <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(32)KB" />
    </work>
</workstage>

<workstage name="64kb">
    <work name="64kb" workers="4" runtime="30">
        <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(4);objects=u(1,8)" />
        <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(64)KB" />
    </work>
</workstage>

```



```

<workstage name="128kb">
  <work name="128kb" workers="1" runtime="30">
    <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(5);objects=u(1,8)" />
    <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(128)KB" />
  </work>
</workstage>

<workstage name="256kb">
  <work name="256kb" workers="1" runtime="30">
    <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(6);objects=u(1,8)" />
    <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(256)KB" />
  </work>
</workstage>

<workstage name="512kb">
  <work name="512kb" workers="1" runtime="30">
    <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(7);objects=u(1,8)" />
    <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(512)KB" />
  </work>
</workstage>

<workstage name="1mb">
  <work name="1mb" workers="1" runtime="30">
    <operation type="read" ratio="80"
config="cprefix=s3test;containers=c(8);objects=u(1,8)" />
    <operation type="write" ratio="20"
config="cprefix=s3test;containers=c(1);objects=u(9,16);sizes=c(1)MB" />
  </work>
</workstage>

<workstage name="cleanup">
  <work type="cleanup" workers="1"
config="cprefix=s3test;containers=r(1,8);objects=r(1,16)" />
</workstage>

<workstage name="dispose">
  <work type="dispose" workers="1" config="cprefix=s3test;containers=r(1,8)" />
</workstage>

```

```
</workflow>

</workload>
```

在 Cosbench Controller Web Console 中，点击 submit 上传上述 workload 文件。

这里我的 Cosbench 出了问题，所有操作全部失败报错 “fail to perform operation”，如图。

```
2018-04-27 03:21:06,575 [ERROR] [AbstractOperator] - worker 1 fail to perform operation s3obstest1/
myobjects1
```

图 10 Cosbench 报错信息

但是 object 文件和 bucket 都被成功创建了。

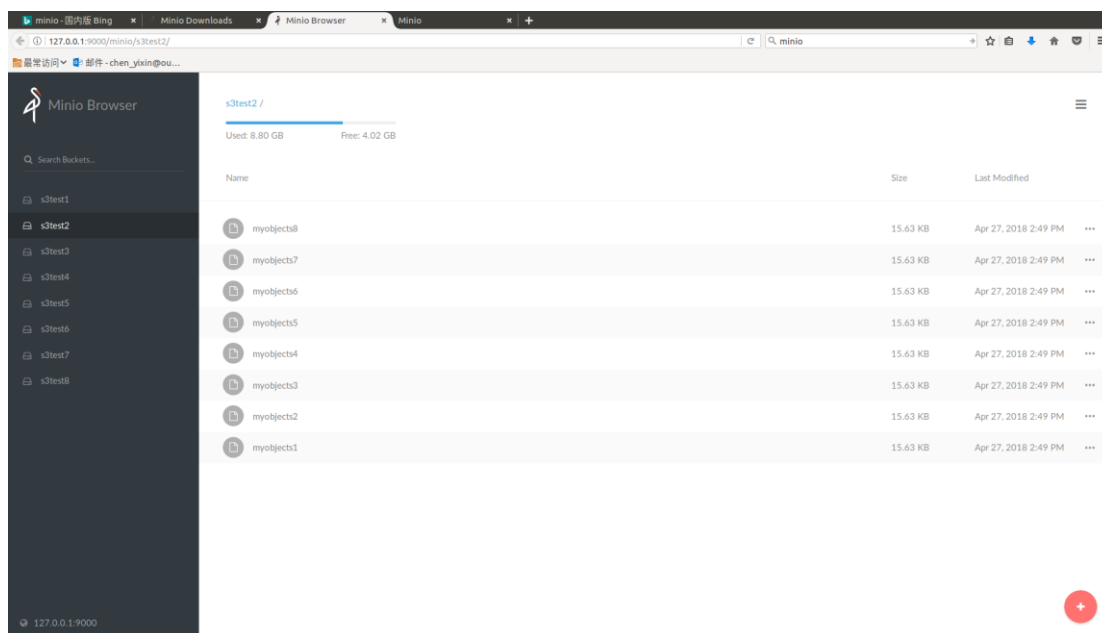


图 11 跑 Cosbench 时 Minio Browser 的显示

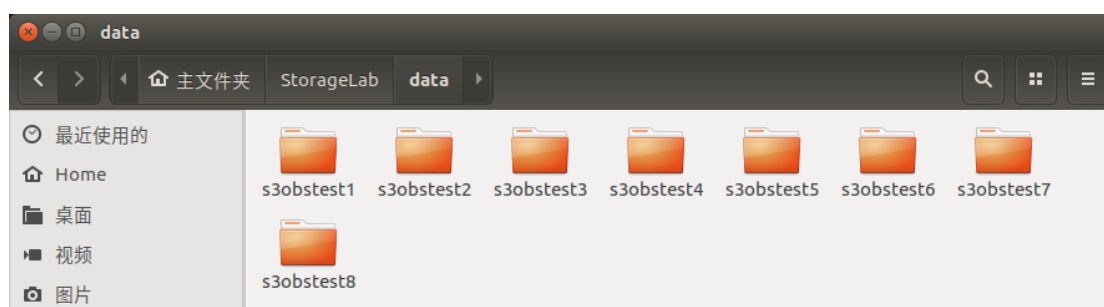


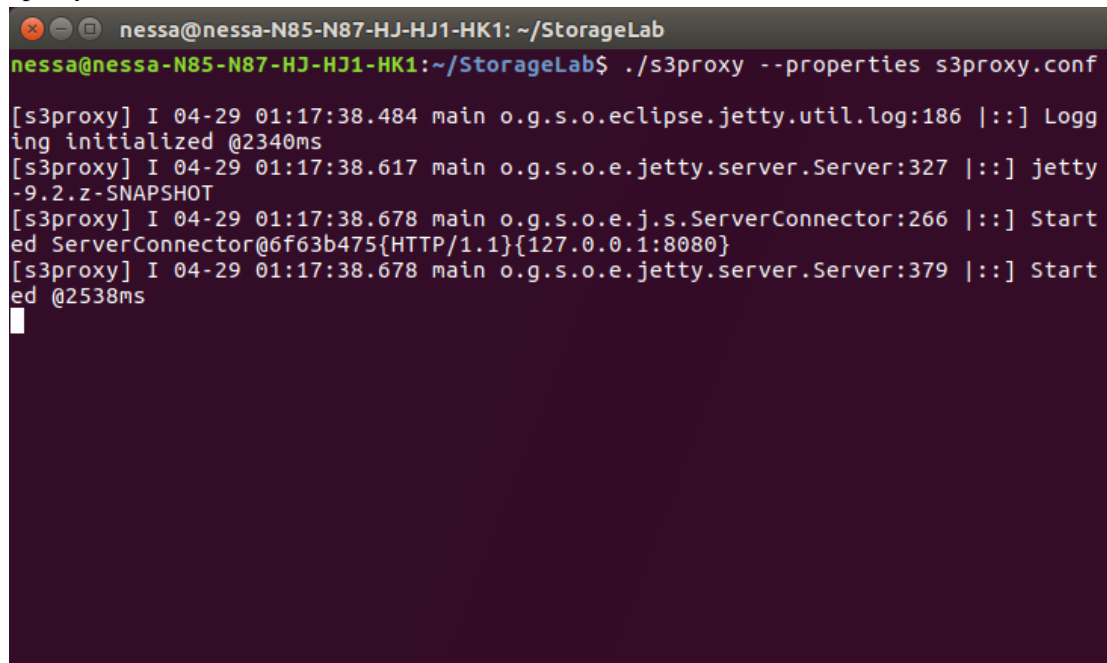
图 12 跑 Cosbench 时本地文件的显示

查阅资料无果（最后也没弄出来真的实在没办法了）。在老师的建议下将 server 改成 s3proxy。

2 s3proxy

Server

在网上下载 release 版 s3proxy, 编辑配置文件 s3proxy.conf, 通过指令 `./s3proxy --properties s3proxy.conf` 运行。

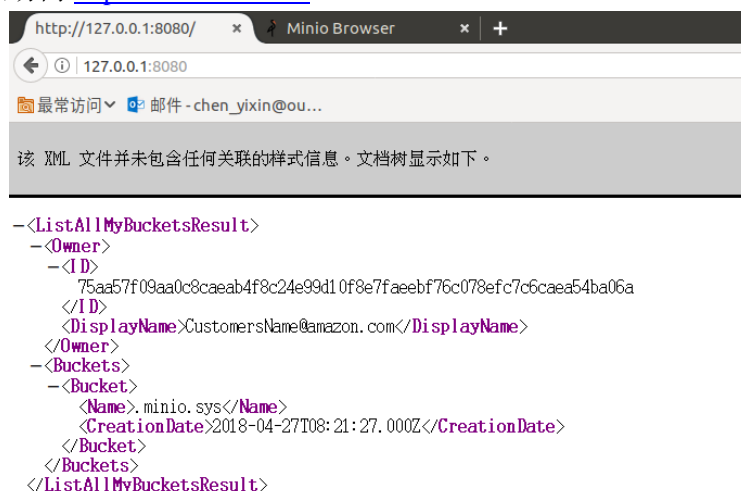


```
nessa@nessa-N85-N87-HJ-HJ1-HK1: ~/StorageLab
nessa@nessa-N85-N87-HJ-HJ1-HK1:~/StorageLab$ ./s3proxy --properties s3proxy.conf

[s3proxy] I 04-29 01:17:38.484 main o.g.s.o.eclipse.jetty.util.log:186 [::] Logging initialized @2340ms
[s3proxy] I 04-29 01:17:38.617 main o.g.s.o.e.jetty.server.Server:327 [::] jetty-9.2.z-SNAPSHOT
[s3proxy] I 04-29 01:17:38.678 main o.g.s.o.e.j.s.ServerConnector:266 [::] Started ServerConnector@6f63b475[HTTP/1.1]{127.0.0.1:8080}
[s3proxy] I 04-29 01:17:38.678 main o.g.s.o.e.jetty.server.Server:379 [::] Started @2538ms
```

图 13 s3proxy 运行终端显示

运行成功后访问 <http://127.0.0.1:8080>。



```
http://127.0.0.1:8080/ x Minio Browser x +
127.0.0.1:8080
最常访问 邮件 - chen_yixin@ou...
该 XML 文件并未包含任何关联的样式信息。文档树显示如下。
- <ListAllMyBucketsResult>
  - <Owner>
    - <ID>
      75aa57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6cae54ba06a
    </ID>
    <DisplayName>CustomersName@amazon.com</DisplayName>
  </Owner>
  - <Buckets>
    - <Bucket>
      <Name>, minio, sys</Name>
      <CreationDate>2018-04-27T08:21:27.000Z</CreationDate>
    </Bucket>
  </Buckets>
</ListAllMyBucketsResult>
```

图 14 s3proxy 网页

Client (aws-shell)

通过 `pip install aws-shell` 安装 aws-shell。之后运行 `aws configure` 命令配置链接, 并通过 `aws configure set default.s3.signature_version s3v4` 配置 s3 链接类型。

```
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$ aws configure
AWS Access Key ID [None]: abcde
AWS Secret Access Key [None]: qazwsxedc
Default region name [None]: s3proxy
Default output format [None]: ENTER
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$ aws configure set default.s3.signature_version s3v4
```

图 15 aws configure

接下来通过 `aws --endpoint-url http://127.0.0.1:8080 s3 ls` 访问 server。

```
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$ aws --endpoint-url http://127.0.0.1:8080 s3 ls
2018-04-27 16:21:27 .minio.sys
nessa@nessa-N85-N87-HJ-HJ1-HK1:~$
```

图 16 访问 s3proxy server

cosbench

修改 workload 文件，将以下部分

```
<storage type="s3"
config="accesskey=V9QRTFZZW4Y0I8LESKSZ;secretkey=PwWkyupV4EoLvR55A6Dvo9B8S
MSfqN41YMKupd;proxyhost=9000;endpoint=http://127.0.0.1:9000" />
```

修改为

```
<storage type="s3"
config="accesskey=abcde;secretkey=qazwsxedc;endpoint=http://127.0.0.1:8080" />
```

并通过 Cosbench Controller Web Console 上传测试文件。Cosbench 成功运行。

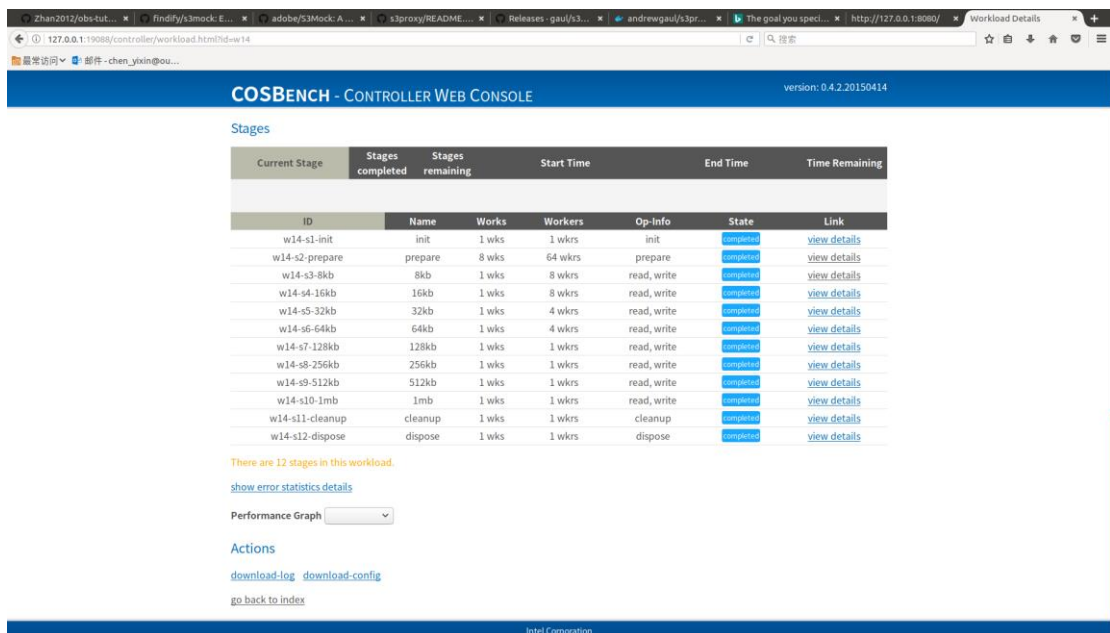


图 17 cosbench 成功运行

点击 Performance Graph 的下拉框，可以查看统计图。

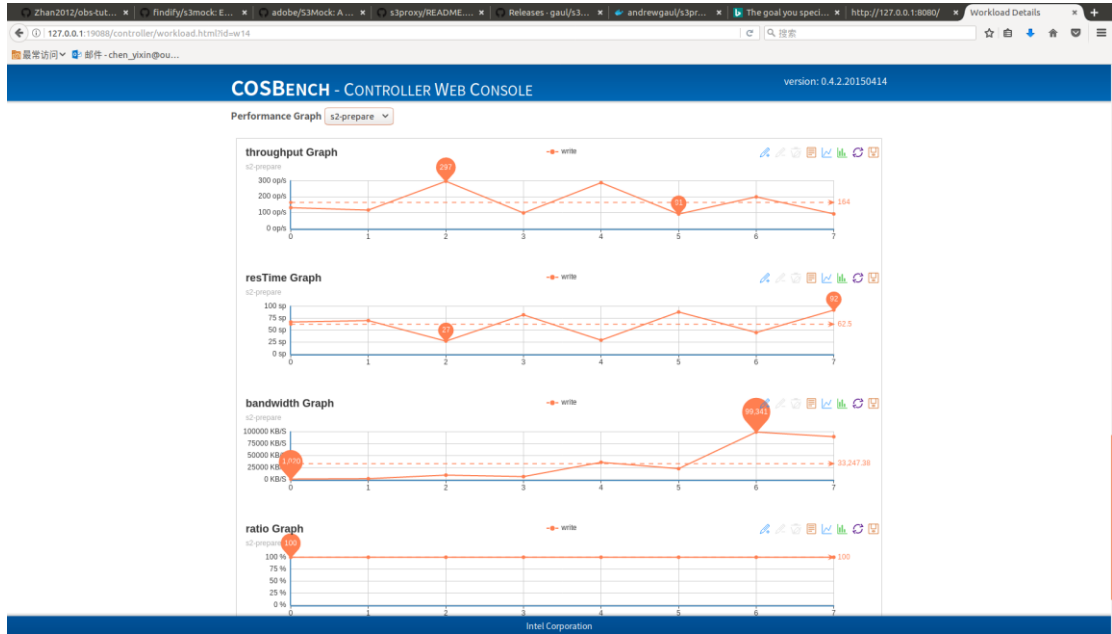


图 18 prepare 阶段

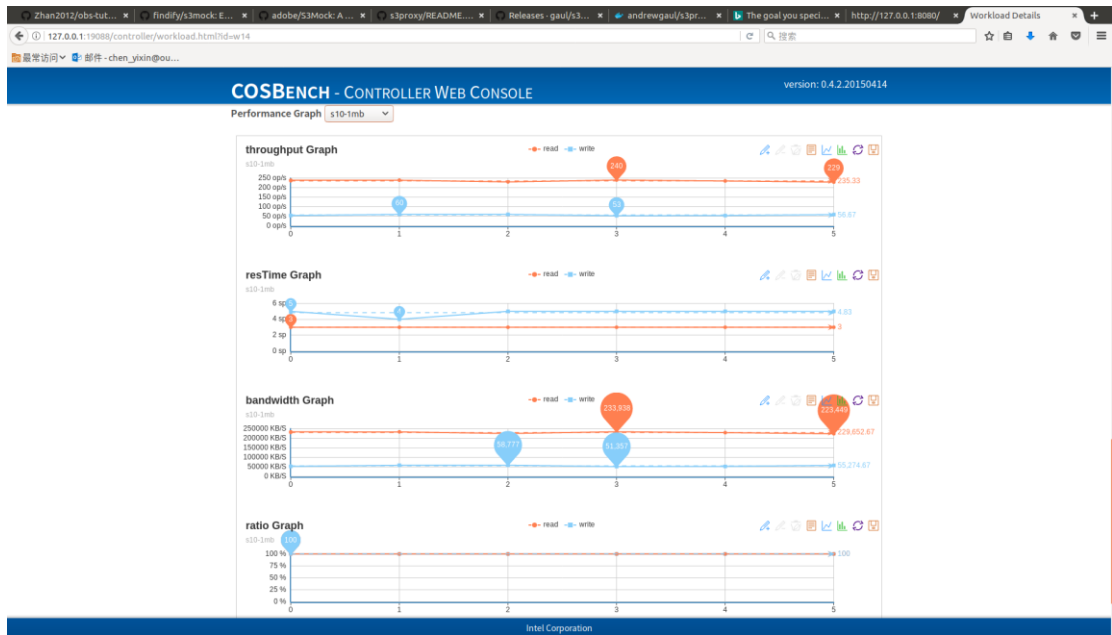


图 19 测试文件大小为 1MB 时

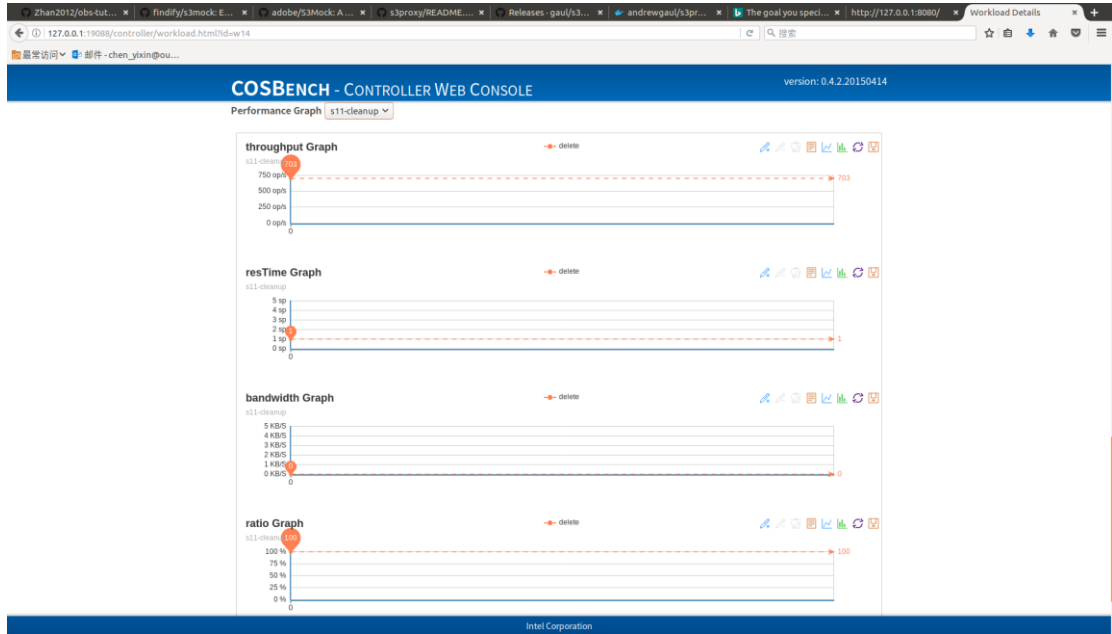


图 20 清空所写入文件时

六、实验总结

这次实验大家都说简单，然而我觉得自己的实验过程真是命途多舛、时运艰难。首先配环境就用了很长时间，期间还有一次一着急删掉了/etc 里的某个文件，什么指令都运行不了了，不得已重装了系统。之后再次配置 java 环境时，所有版本都对了，和同学的配置过程完全一样，但是 cosbench 就是跑不起来，愁死个人。之后在老师的建议下，我尝试用 docker 运行 cosbench，才解决了这个问题。之后下一个问题又来了，如我在实验过程中写的那样，cosbench 的 workloads 明明可以成功操作 Minio Server 的文件，但是所有的测试结果全都是 failed，十分奇怪。我为解决这个问题查谷歌查到掉头发，找到了一个 cosbench 的论坛，翻遍了里面所有的 user 提问，只找到一个和我现象差不多的，然而没有人回答他……后来又在老师的建议下，改用 s3proxy，cosbench 果然跑通了。这之后我又重新在 Minio Server 上跑了 cosbench 测试，然而还是和之前一样失败了，最终也没搞懂为什么。大家都说简单的一个实验，我居然拖到 deadline 才交，我想这说明我的知识水平还有待提高。

这次实验非常感谢老师的耐心答疑。当我绝望之时，是老师给了我希望之火，指引我抛弃 minio 转向 s3proxy，并且教给了我在这种事情一团糟的状况下，逐步排除可能性，最终解决问题的思路。我想我还有很多要学习的地方。

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.