

华中科技大学计算机科学与技术学院

大数据处理实践实验报告

计科校交 1801

车春池

U201816030

TOC

- 引言
- 实验目的
- 实验环境
- 实验内容及步骤
- 实验小结
- 附录

引言

大数据处理平台 Hadoop 是 Apache 基于 Google 的三篇论文《分布式文件系统》, 《分布式计算模型》和《Bigtable》的开源系统框架。Hadoop 基本框架包含 HDFS, YARN, MapReduce 编程模型基础类, 具体涉及到大数据文件在分布式存取过程中的数据切片, 数据组织, 副本管理, 一致性保证, 复制策略和读写访问等, 其中各功能模块参数对分布式文件系统的 I/O 性能及分布式大数据处理性能影响较大。

实验目的

本实验基于 Docker 容器模拟 4 台机器搭建 Hadoop 平台，从黑盒测试和使用角度了解大数据处理系统的模块结构和运行原理，通过比较本机文件系统结构，加深对 HDFS 中系统框架，组件功能，数据一致性，故障容错和副本策略等的理解，以达到以下目的：

- 通过运行脚本和进程状态了解分布式文件系统和分布式处理系统的系统结构和功能模块，并能对系统故障进行故障定位排除；
- 了解 Hadoop 分布式文件系统的结构及其在本地文件系统的映射结构，加深对 Hadoop 集群，数据节点，数据版本一致性，数据块，副本，元数据等的理解；
- 了解副本数量对实际存储空间的影响及副本在数据节点上复制的顺序；
- 了解 HDFS 数据块大小对不同大小的数据文件在 HDFS 系统存取性能的影响；
- 了解 HDFS 故障容错机制及其对数据一致性的影响；
- 了解副本复制的机架感知策略配置方法；
- 了解 Hadoop 核心组件参数对 HDFS 存取性能的影响；
- 自行设计 HDFS 黑盒试验加深对 HDFS 原理的影响；

实验环境

阿里云服务器

- CPU：双核
- 内存：4 GiB
- OS：Ubuntu 20.04 64 bit

实验内容及步骤

实验1-1 Hadoop 基本环境搭建及组件进程熟悉

拉取镜像

```
docker pull daocloud.io/library/centos:latest
```

查看镜像是否下载成功

```
docker image list 查看本机上存在的镜像列表。
```

创建固定网络，最好与本机不属于同一个网段

```
docker network create --subnet=10.0.0.0/16 netgroup
```

配置 1 主 3 从集群节点IP

cluster-master

```
docker run -d -p 18088:18088 -p 9870:9870 --privileged -ti -v /sys/fs/cgroup:/sys/fs/cgroup --name cluster-master -h cluster-master --net netgroup --ip 10.0.0.2 daocloud.io/library/centos /usr/sbin/init
```

cluster-slaves

```
docker run -d --privileged -ti -v /sys/fs/cgroup:/sys/fs/cgroup --name cluster-slave-1 -h cluster-slave-1 --net netgroup --ip 10.0.0.3 daocloud.io/library/centos /usr/sbin/init
```

```
docker run -d --privileged -ti -v /sys/fs/cgroup:/sys/fs/cgroup --name cluster-slave-2 -h cluster-slave-2 --net netgroup --ip 10.0.0.4 daocloud.io/library/centos /usr/sbin/init
```

```
docker run -d --privileged -ti -v /sys/fs/cgroup:/sys/fs/cgroup --name cluster-slave-3 -h cluster-slave-3 --net netgroup --ip 10.0.0.5 daocloud.io/library/centos /usr/sbin/init
```

启动控制台并进入容器

```
docker exec -it cluster-master /bin/bash
```

```
docker exec -it cluster-slave-1 /bin/bash
```

```
docker exec -it cluster-slave-2 /bin/bash
```

```
docker exec -it cluster-slave-3 /bin/bash
```

安装openssh

```
yum -y install openssh openssh-server openssh-clients
```

启动ssh服务

```
systemctl start sshd
```

设置 ssh

```
vi /etc/ssh/sshd_config
```

将原来的 StrictHostKeyChecking ask 设置为 StrictHostKeyChecking no，然后去掉注释，保存。

重新启动ssh服务

```
systemctl restart sshd
```

集群中所有机器执行安装和启动 ssh

在所有客户机上运行：

```
yum -y install openssh openssh-server openssh-clients  
systemctl start sshd
```

设置免密登录

安装 passwd，修改密码

```
yum -y install passwd  
passwd  
  
ssh-keygen -t rsa  
#一路回车  
cat ~/.ssh/id_rsa.pub > ~/.ssh/authorized_keys  
  
#分发密钥文件，免密登录  
ssh root@cluster-slave-1 'mkdir ~/.ssh'  
scp ~/.ssh/authorized_keys root@cluster-slave-1:~/ssh  
ssh root@cluster-slave2 'mkdir ~/.ssh'  
scp ~/.ssh/authorized_keys root@cluster-slave-2:~/ssh  
ssh root@cluster-slave3 'mkdir ~/.ssh'  
scp ~/.ssh/authorized_keys root@cluster-slave-3:~/ssh
```

安装 ansible 用于分发

```
yum -y install epel-release  
yum -y install ansible
```

修改ansible集群配置

```
vi /etc/ansible/hosts
```

```
[cluster]  
cluster-master  
cluster-slave-1  
cluster-slave-2
```

```
cluster-slave-3
```

```
[master]  
cluster-master
```

```
[slaves]  
cluster-slave-1  
cluster-slave-2  
cluster-slave-3
```

追加配置，防止被重写

```
vi ~/.bashrc

:>/etc/hosts
cat >>/etc/hosts<<EOF
127.0.0.1 localhost
10.0.0.2 cluster-master
10.0.0.3 cluster-slave-1
10.0.0.4 cluster-slave-2
10.0.0.5 cluster-slave-3
EOF

source ~/.bashrc
```

分发配置

```
ansible cluster -m copy -a "src=~/ .bashrc dest=~/"
```

分发安装 java sdk

```
ansible cluster -m yum -a "name=java-1.8.0-openjdk,java-1.8.0-openjdk-devel state=latest"
```

安装hadoop安装包，在清华镜像选择版本，下载至 /opt 目录下，解压并建立链接

```
cd /opt
wget http://mirrors.tuna.tsinghua.edu.cn/apache/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz
tar -xvzf hadoop-3.2.1.tar.gz
ln -s hadoop-3.2.1 hadoop
```

设置 JAVA 和 HADOOP 环境

```
vi ~/.bashrc

# hadoop
export HADOOP_HOME=/opt/hadoop
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH

#j ava
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.262.b10-
0.el8_2.x86_64 #注意版本和目录
export PATH=$JAVA_HOME/bin:$PATH
```

设置 Hadoop 配置文件

下面这四个文件在下载的 Hadoop 目录的 ./etc/hadoop/ 目录下。

core-site.xml

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/home/hadoop/tmp</value>
    <description>A base for other temporary directories.
  </description>
  </property>
  <!-- file system properties -->
  <property>
    <name>fs.default.name</name>
    <value>hdfs://cluster-master:9000</value>
  </property>
  <property>
    <name>fs.trash.interval</name>
    <value>4320</value>
  </property>
</configuration>
```

hdfs-site.xml

```
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/home/hadoop/tmp/dfs/name</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>/home/hadoop/data</value>
```

```
</property>
<property>
  <name>dfs.replication</name>
  <value>3</value>
</property>
<property>
  <name>dfs.webhdfs.enabled</name>
  <value>true</value>
</property>
<property>
  <name>dfs.permissions.superusergroup</name>
  <value>staff</value>
</property>
<property>
  <name>dfs.permissions.enabled</name>
  <value>false</value>
</property>
```

mapred-site.xml

```
<configuration>
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
<property>
  <name>mapred.job.tracker</name>
  <value>cluster-master:9001</value>
</property>
<property>
  <name>mapreduce.jobtracker.http.address</name>
  <value>cluster-master:50030</value>
</property>
<property>
  <name>mapreduce.jobhistory.address</name>
  <value>cluster-master:10020</value>
</property>
<property>
  <name>mapreduce.jobhistory.webapp.address</name>
  <value>cluster-master:19888</value>
</property>
<property>
  <name>mapreduce.jobhistory.done-dir</name>
  <value>/jobhistory/done</value>
</property>
<property>
  <name>mapreduce.intermediate-done-dir</name>
```

```
<value>/jobhistory/done_intermediate</value>
</property>
<property>
  <name>mapreduce.job.ubertask.enable</name>
  <value>true</value>
</property>
</configuration>
```

yarn-site.xml

```
<configuration>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>cluster-master</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>cluster-master:18040</value>
  </property>
  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>cluster-master:18030</value>
  </property>
  <property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>cluster-master:18025</value>
  </property> <property>
    <name>yarn.resourcemanager.admin.address</name>
    <value>cluster-master:18141</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address</name>
    <value>cluster-master:18088</value>
  </property>
  <property>
    <name>yarn.log-aggregation-enable</name>
    <value>true</value>
  </property>
  <property>
```

```
<name>yarn.log-aggregation.retain-seconds</name>
<value>86400</value>
</property>
<property>
  <name>yarn.log-aggregation.retain-check-interval-seconds</name>
  <value>86400</value>
</property>
<property>
  <name>yarn.nodemanager.remote-app-log-dir</name>
  <value>/tmp/logs</value>
</property>
<property>
  <name>yarn.nodemanager.remote-app-log-dir-suffix</name>
  <value>logs</value>
</property>
</configuration>
```

修改 start-dfs.sh 和 stop-dfs.sh 文件

在文件头部增加：

```
#!/usr/bin/env bash
HDFS_DATANODE_USER=root
HADOOP_SECURE_DN_USER=hdfs
HDFS_NAMENODE_USER=root
HDFS_SECONDARYNAMENODE_USER=root
```

修改 start-yarn.sh 和 stop-yarn.sh 文件

在文件头部增加：

```
#!/usr/bin/env bash
YARN_RESOURCEMANAGER_USER=root
HADOOP_SECURE_DN_USER=yarn
YARN_NODEMANAGER_USER=root
```

分发到客户机

```
tar -cvf hadoop-dis.tar hadoop hadoop-3.2.1

vi hadoop-dis.yaml
---
- hosts: cluster
  tasks:
```

```
- name: copy .bashrc to slaves
  copy: src=~/bashrc dest=~
  notify:
    - exec source
- name: copy hadoop-dis.tar to slaves
  unarchive: src=/opt/hadoop-dis.tar dest=/opt

handlers:
- name: exec source
  shell: source ~/bashrc
```

开始分发

```
ansible-playbook hadoop-dis.yaml
```

格式化 namenode

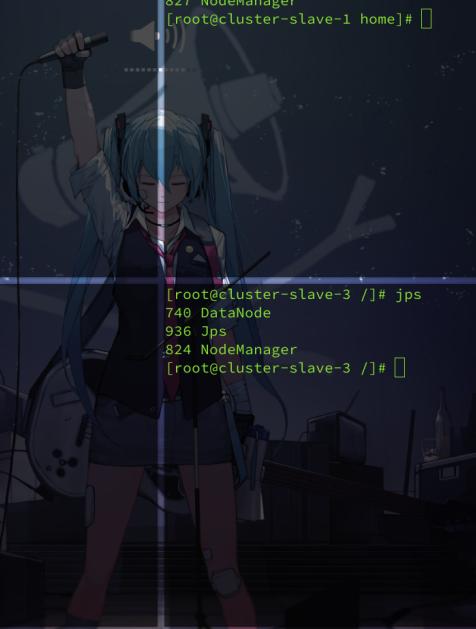
```
hadoop namenode -format
```

启动集群

```
cd $HADOOP_HOME/sbin
./start-all.sh
```

实验 1-1 结果

检查 Hadoop 的 Java 进程：



```
Starting namenodes on [cluster-master]
Last login: Fri Oct  9 13:08:17 UTC 2020 on pts/1
Starting datanodes
Last login: Fri Oct  9 13:13:38 UTC 2020 on pts/1
Starting secondary namenodes [cluster-master]
Last login: Fri Oct  9 13:13:41 UTC 2020 on pts/1
Starting resourcemanager
Last login: Fri Oct  9 13:13:46 UTC 2020 on pts/1
Starting nodemanagers
Last login: Fri Oct  9 13:13:55 UTC 2020 on pts/1
[root@cluster-master hadoop]# jps
3235 DataNode
3684 ResourceManager
4117 Jps
3834 NodeManager
3435 SecondaryNameNode
3087 NameNode
[root@cluster-master hadoop]# 

[root@cluster-slave-1 ~]# jps
950 Jps
743 DataNode
827 NodeManager
[root@cluster-slave-1 ~]# 

[root@cluster-slave-2 ~]# jps
720 DataNode
803 NodeManager
847 Jps
[root@cluster-slave-2 ~]# 

[root@cluster-slave-3 ~]# jps
740 DataNode
936 Jps
824 NodeManager
[root@cluster-slave-3 ~]# 
```

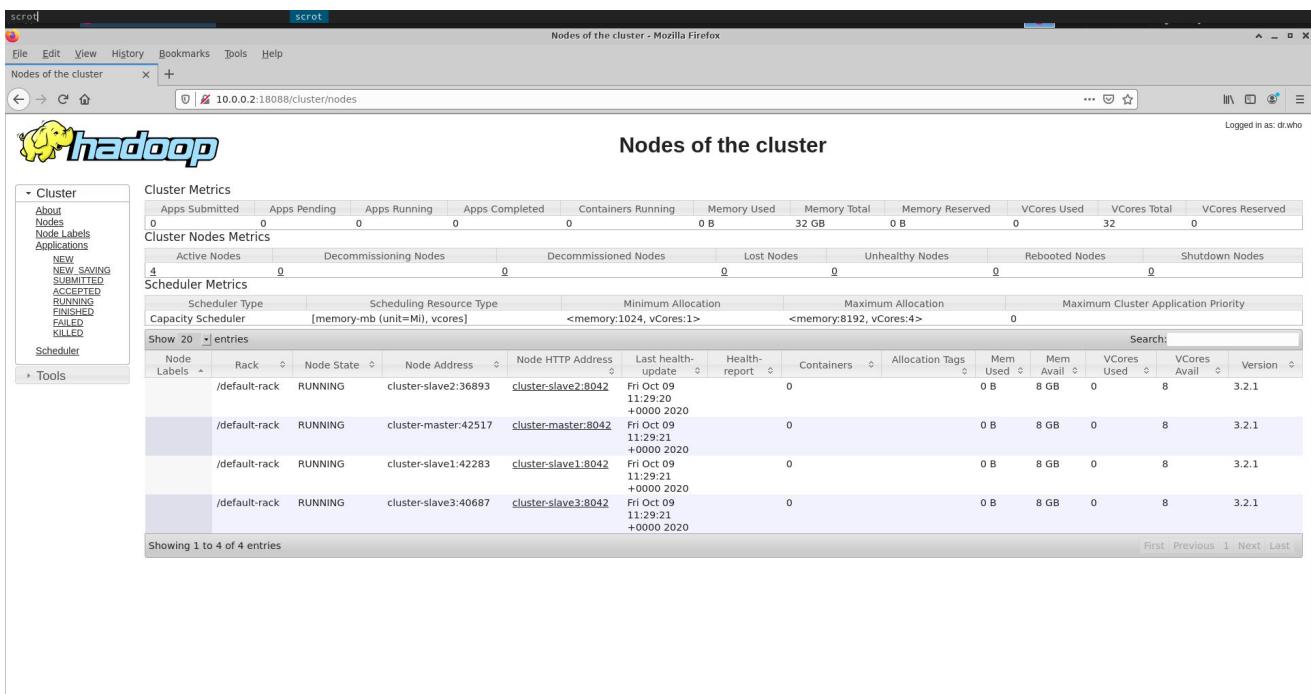
可以看到主节点和从节点均正确启动。

访问验证

在本地浏览器里面打开下面两个网址：

```
http://10.0.0.2:18088
http://10.0.0.2:9870
```

结果如下：



The screenshot shows the 'Nodes of the cluster' interface of a Hadoop cluster. The left sidebar has sections for Cluster Metrics, Cluster Nodes Metrics, Scheduler Metrics, and Tools. The main content area shows the following data:

Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Allocation Tags	Mem Used	Mem Avail	VCores Used	VCores Avail	Version
/default-rack	RUNNING	cluster-slave2:36893	cluster-slave2:8042	Fri Oct 09 11:29:20 +0000 2020	0	0	0 B	8 GB	0	8	3.2.1		
/default-rack	RUNNING	cluster-master:42517	cluster-master:8042	Fri Oct 09 11:29:21 +0000 2020	0	0	0 B	8 GB	0	8	3.2.1		
/default-rack	RUNNING	cluster-slave1:42283	cluster-slave1:8042	Fri Oct 09 11:29:21 +0000 2020	0	0	0 B	8 GB	0	8	3.2.1		
/default-rack	RUNNING	cluster-slave3:40687	cluster-slave3:8042	Fri Oct 09 11:29:21 +0000 2020	0	0	0 B	8 GB	0	8	3.2.1		

在浏览器可以看到 4 个节点都在正常运行。

实验 1-2 本地文件系统与 HDFS 文件系统操作及组织结构

创建目录并上传文件

```
hadoop fs -mkdir /input  
hadoop fs -ls /  
hadoop fs -put [file-to-be-upload] /input  
hadoop fs -ls /input
```

结果如下：



```
Starting datanodes  
Last login: Fri Oct  9 13:13:38 UTC 2020 on pts/1  
Starting secondary namenodes [cluster-master]  
Last login: Fri Oct  9 13:13:41 UTC 2020 on pts/1  
Starting resourcemanager  
Last login: Fri Oct  9 13:13:46 UTC 2020 on pts/1  
Starting nodemanagers  
Last login: Fri Oct  9 13:13:55 UTC 2020 on pts/1  
[root@cluster-master hadoop]# jps  
3235 DataNode  
3684 ResourceManager  
4117 Jps  
3834 NodeManager  
3435 SecondaryNameNode  
3087 NameNode  
[root@cluster-master hadoop]# clear  
  
[root@cluster-master hadoop]# hadoop fs -ls /  
[root@cluster-master hadoop]# hadoop fs -mkdir /data  
[root@cluster-master hadoop]# hadoop fs -ls /  
Found 1 items  
drwxr-xr-x  - root staff          0 2020-10-09 13:16 /data  
[root@cluster-master hadoop]# hadoop fs -put ../hadoop-dis.tar /data  
2020-10-09 13:17:17,546 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false  
2020-10-09 13:17:23,328 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false  
2020-10-09 13:17:28,665 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false  
2020-10-09 13:17:33,634 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false  
2020-10-09 13:17:38,589 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false  
2020-10-09 13:17:43,341 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false  
2020-10-09 13:17:48,311 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false  
[root@cluster-master hadoop]# hadoop fs -ls /data  
Found 1 items  
-rw-r--r--  3 root staff  914688000 2020-10-09 13:17 /data/hadoop-dis.tar  
[root@cluster-master hadoop]#
```

观察并分析本地文件系统的目录和 HDFS 文件系统目录的关系

观察如下：

```
[root@cluster-master hadoop]# hadoop fs -ls /
Found 1 items
drwxr-xr-x  - root staff          0 2020-10-10 01:22 /input
[root@cluster-master hadoop]# hadoop fs -ls /input
Found 1 items
-rw-r--r--  3 root staff  915056640 2020-10-10 01:22 /input/hadoop-dis.tar
[root@cluster-master hadoop]# ls data/current/
BP-359623114-10.0.0.2-1602292758436 VERSION
[root@cluster-master hadoop]# ls -lht data/current/BP-359623114-10.0.0.2-1602292758436/current/finalized/subdir0/subdir0/
total 880M
-rw-r--r-- 1 root root 105M Oct 10 01:22 blk_1073741831
-rw-r--r-- 1 root root 838K Oct 10 01:22 blk_1073741831_1007.meta
-rw-r--r-- 1 root root 128M Oct 10 01:22 blk_1073741830
-rw-r--r-- 1 root root 1.1M Oct 10 01:22 blk_1073741830_1006.meta
-rw-r--r-- 1 root root 128M Oct 10 01:22 blk_1073741829
-rw-r--r-- 1 root root 1.1M Oct 10 01:22 blk_1073741829_1005.meta
-rw-r--r-- 1 root root 128M Oct 10 01:22 blk_1073741828
-rw-r--r-- 1 root root 1.1M Oct 10 01:22 blk_1073741828_1004.meta
-rw-r--r-- 1 root root 128M Oct 10 01:22 blk_1073741827
-rw-r--r-- 1 root root 1.1M Oct 10 01:22 blk_1073741827_1003.meta
-rw-r--r-- 1 root root 128M Oct 10 01:22 blk_1073741826
-rw-r--r-- 1 root root 1.1M Oct 10 01:22 blk_1073741826_1002.meta
-rw-r--r-- 1 root root 128M Oct 10 01:21 blk_1073741825
-rw-r--r-- 1 root root 1.1M Oct 10 01:21 blk_1073741825_1001.meta
[root@cluster-master hadoop]# ls -lht tmp/dfs/name/current/
total 3.1M
-rw-r--r-- 1 root root 1.0M Oct 11 02:45 edits.inprogress_00000000000000000030
-rw-r--r-- 1 root root  3 Oct 11 02:45 seen_txid
-rw-r--r-- 1 root root 212 Oct 11 02:45 VERSION
-rw-r--r-- 1 root root  62 Oct 11 02:45 fsimage_00000000000000000029.md5
-rw-r--r-- 1 root root  668 Oct 11 02:45 fsimage_00000000000000000029
-rw-r--r-- 1 root root  62 Oct 10 01:32 fsimage_00000000000000000028.md5
-rw-r--r-- 1 root root  668 Oct 10 01:32 fsimage_00000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:31 edits_00000000000000000027-00000000000000000028
-rw-r--r-- 1 root root  42 Oct 10 01:31 edits_00000000000000000027-00000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:22 edits_00000000000000000021-00000000000000000026
[root@cluster-master hadoop]# ls -lht tmp/dfs/namesecondary/current/
total 1.1M
-rw-r--r-- 1 root root 212 Oct 10 01:32 VERSION
-rw-r--r-- 1 root root 668 Oct 10 01:32 fsimage_00000000000000000028
-rw-r--r-- 1 root root  62 Oct 10 01:32 fsimage_00000000000000000028.md5
-rw-r--r-- 1 root root  42 Oct 10 01:32 edits_00000000000000000027-00000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:32 edits_00000000000000000021-00000000000000000026
-rw-r--r-- 1 root root  62 Oct 10 01:32 fsimage_00000000000000000000000000000000.md5
-rw-r--r-- 1 root root  394 Oct 10 01:32 fsimage_00000000000000000000000000000000
[root@cluster-master hadoop]#
```

结合 HDFS 原理分析如下：

- HDFS 基于本地文件系统，并通过操作本地文件系统来存储数据
- Hadoop 对本地文件系统封装了一个抽象层，抽象出了 Hadoop 文件系统 HDFS
- 为了提供统一的数据访问接口，Hadoop 参考了 Linux 的虚拟文件系统实现，引入了 HDFS 抽象文件系统，并提供了大量实现
- 在上面图中可以看到，本地文件系统里面存储了元数据，比如文件名以blk开头的文件
- 在 NameNode 目录下可以看到这里存储了元数据文件，包括fsimage, editlog, seen_txid等
- fsimage文件其实是 Hadoop 文件系统元数据的一个永久性的检查点，其中包含 Hadoop 文件系统中的所有目录和文件 idnode 的序列化信息
- edits文件存放的是 Hadoop 文件系统的所有更新操作的路径，文件系统客户端执行的所以写操作首先会被记录到 edits 文件中
- VERSION文件是 Java 属性文件
- 同时可以看到 SecondaryNameNode 目录下没有seen_txid文件
- 总的来说，HDFS 是比本地文件系统更高层次的文件系统的抽象，将多台机器组成的文件系统看成一个逻辑上的整体

实验 1-3 Hadoop 故障及容错恢复

在这个实验中，我写了三个脚本，来帮助我观察 HDFS 目录及本地目录的动态变化，分别是：

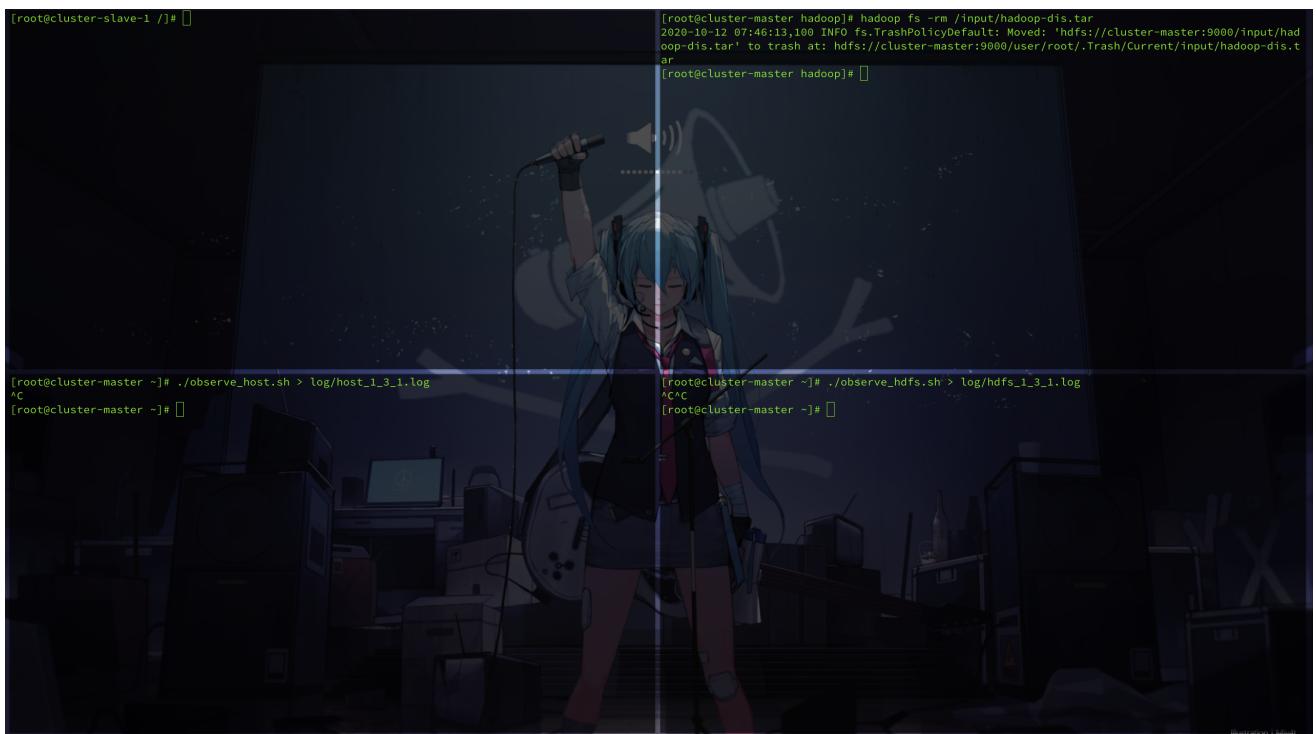
- observe_host.sh：用于监测主节点本地目录的动态变化
- observe_hdfs.sh：用于监测 HDFS 目录的动态变化
- observe_error.sh：用于故障恢复实验监测主节点和从节点及 HDFS 目录的动态变化

脚本的具体内容将会在附录里给出。

NameNode 主备实验

- 停止 SecondaryNameNode
- 在 HDFS 上进行文件上传和删除操作
- 对比 NameNode 和 SecondaryNameNode 目录的变化
- 监测 HDFS 目录动态变化
- 停止 NameNode
- 监测 HDFS 目录动态变化
- 启动 SecondaryNameNode
- 监测 HDFS 目录动态变化

实验过程截图：



实验数据由 observe_host.sh 和 observe_hdfs.sh 脚本文件输出重定向到 .log 文件里面，分别有：

- host_1_3_1.log (停止 SecondaryNameNode 后上传文件过程本地目录的变化)
- host_1_3_2.log (停止 SecondaryNameNode 后删除文件过程本地目录的变化)
- hdfs_1_3_1.log (停止 SecondaryNameNode 后上传文件过程 HDFS 目录的变化)

- hdfs_1_3_2.log (停止 SecondaryNameNode 后删除文件过程 HDFS 目录的变化)
- hdfs_1_3_3.log (停止 NameNode HDFS 目录的动态变化)
- hdfs_1_3_4.log (启动 SecondaryNameNode HDFS 目录的动态变化)

这些 .log 文件的具体内容会在附录给出。

下面对实验数据进行分析：

- 停止 SecondaryNameNode 后本地目录的 NameNode 目录中以 edit 开头的文件发生了变化。
(根据文件最近修改的时间可以看到)
- 在上传文件的过程中会可以发现在 HDFS 目录下增加了目标文件，并且文件大小不断增加
- 在删除文件过程中文件不是直接从 HDFS 目录中消失，而是会转存在 /user/.trash/ 目录中

DataNode 故障恢复实验

- 在主节点删除实验 1-2 中上传第 1 步中上传的大文件
- 删除过程中，快速随机制造故障，关停从节点
- 监测主节点和从节点和 HDFS 目录的动态变化
- 恢复启动之前关停节点，监测主节点和从节点的 HDFS 目录的动态变化
- 等待一段时间后，监测主节点和从节点和 HDFS 目录的动态变化

实验数据由 observe_error.sh 脚本文件输出重定向到 .log 文件里面，分别有：

- error_test_1.log
- error_test_2.log
- error_test_3.log

这些 .log 文件的具体内容将会在附录给出。

下面分析总结一下 Hadoop 分布式文件系统的一致性和故障容错恢复机制。

一致性：

- HDFS 采用简单的一致性模型，HDFS 应用需要一个“一次写入多次读取”的文件访问模型。一个文件经过创建、写入和关闭之后就不需要改变。
- Namenode 启动后会进入一个称为安全模式的特殊状态。处于安全模式的Namenode 是不会进行数据块的复制的。Namenode 从所有的 Datanode 接收心跳信号和块状态报告。
- Namenode 在内存中保存着整个文件系统的名字空间和文件数据块映射(Blockmap) 的映像。

故障容错恢复：

- 故障类型包括三类故障：节点失败，网络故障，数据损坏
- 故障检测机制有：节点失败检测机制，通信故障检测机制，数据错误检测机制
- 写容错：每个 DN 收到数据包后会返回确认应答，如果 NN 没有收到某个 DN 的确认，就会认为这个 DN 挂了，然后调整路线跳过该 DN
- 读容错：当用户询问数据块的地址时，NN 发送所有 DN 的位置，如果某个 DN 存在故障，那么就可以通过列表从其他 DN 读取数据
- 数据节点失效：系统会维护两张表，数据块列表和 DN 列表，如果发现某个 DN 上的数据块存在错误，系统会从数据块列表中删除此数据块，如果发现是 DN 失效，系统会同时更

新这两张表

- 系统周期性扫描第一张表，针对那些没有按要求的数量进行复制的数据块，系统会找另外的 DN 从已备份此数据块的 DN 中进行复制
- HDFS 的备份规则。

实验 1-4 Hadoop 参数配置对性能的影响

在这个实验中，我写了一个 top_record.sh 脚本来监测 top 输出的动态变化，这个脚本的具体内容也会在附录中给出。

- 修改 hdfs-site.xml 中参数，观察读写性能
- 将 dfs.block.size 分别设置成为 512KB，16M，32M，64M，128M，256M 的大小，拷贝删除文件
- 使用 top 统计在 docker 节点的 CPU 和内存变化

通过在 hdfs-site.xml 文件中添加下面几行来修改 dfs.block.size 的值：

```
<property>
    <name>dfs.block.size</name>
    <value>size</value>
</property>
```

实验数据由 top_record.sh 脚本文件输出重定向到 .log 文件，分别有：

- top_1M.log(dfs.block.size = 1M)
- top_100M.log(dfs.block.size = 100M)
- top_500M.log(dfs.block.size = 500M)

这些 .log 文件的具体内容将会在附录给出。

通过实验数据分析一下 Hadoop 参数配置对性能的影响：

下面表格列出一些 hdfs-site.xml 文件中参数的含义：

参数名称	含义
dfs.name.dir	指定 name 镜像文件存放目录，默认为 core-site 中配置的 tmp 目录
dfs.data.dir	真正的 datanode 数据保存路径，可以写多块硬盘，中间以逗号分隔
dfs.replication	hdfs 数据块的复制份数，默认 3
dfs.permissions	是否需要角色权限验证

参数名称	含义
dfs.block.size	每个文件块的大小，默认是 64M
dfs.namenode.handler.count	NameNode 节点上面为处理 datanode 节点来启动的远程调用的服务线程数量
dfs.datanode.max.xcievers	相当于 linux 下的打开文件最大数量
dfs.datanode.handler.count	datanode 节点上为处理 datanode 节点的远程调用开启的服务线程数量，默认为 3

- 通过 top_x.log 文件中的数据，我们可以看到拷贝删除文件过程中 java 进程，也就是 hadoop 进程占很高 CPU 资源
- 通过对比三个 top_x.log 脚本，发现 dfs.block.size 设置得较大，耗费的 CPU 资源会相对提高，但同时 Hadoop 性能会有所提高
- 在网络较好的集群中，建议将这个参数提升，以获得更高的性能

实验 1-5 HDFS 副本复制策略，流程及性能分析

HDFS 副本的复制策略

在之前的实验过程中，已经在 error_test_x.log 文件里面记录了主节点和 3 个从节点的空间占用大小变化。这些 .log 文件可以在附录里面找到。

下面分析一下 HDFS 副本的复制策略：

- 第一个副本随机选择，但是不会选择存储过满的节点
- 第二个副本放在和第一个副本不同且随机选择的机架上
- 第三个和第二个放在同一个机架上的不同节点上
- 剩余副本随机存放
- 流水线复制

实验 1-6 自行设计 Hadoop 黑盒性能实验案例

dfs.replication 参数对 Hadoop 性能的影响

通过运行不同 dfs.replication 的 Hadoop 来测试该参数对 Hadoop 性能的影响。

测试流程和实验 1-4 基本一致，我设置了 dfs.replication 为 4,7,9（默认是 3），实验数据通过 top.record.sh 脚本重定向输出到 top_1_6_4.log,top_1_6_7.log,top_1_6_9.log 文件中，与实验 1-4 中的实验数据进行对比。这些 .log 文件可以在附录找到。

对实验数据进行分析：

- 首先 dfs.replication 这个参数是个 client 参数，即 node level 参数。需要在每台 datanode 上设置
- 实验结果表明这个参数设置太大对 hadoop 性能基本没影响
- 一个文件，上传到 HDFS 上时指定的是几个副本就是几个，后续修改副本数对已经上传的文件不会有影响
- 该实验中只有 3 个 datanode，如果指定副本数大于 3，是不会生效的，因为每个 datanode 上只能存放一个副本
- 所以说这个参数对 Hadoop 性能影响不大

实验小结

通过本次实验，我对 Hadoop 这个分布式文件系统平台有了更深一步的了解。

通过实验 1-1 和 1-2，我亲身体会了 Hadoop 大数据平台的搭建过程，对 Hadoop 有了一个更清晰的总体了解。

通过实验 1-3 至 1-6，我通过黑盒测试的角度从不同方面对 Hadoop 的一致性，容错机制，副本复制策略和 HDFS 存取性能等有了更深刻的了解，另外还对分布式运维部署等概念有了初步认识。

此外在实验的过程中我通过写 Shell 脚本的方式对实验过程进行动态监测，这是我熟悉 Linux 和系统编程能力的体现。

总的来说，整个实验过程虽曲折但不乏有趣，既锻炼了我的动手能力也加深了我对课上知识的理解，受益匪浅。

附录

脚本文件

observe_host.sh

```
#!/bin/bash
# Shell for observe /home/hadoop/tmp/dfs/name/,
#/home/hadoop/tmp/dfs/name/current, /home/hadoop/tmp/dfs/namesecondary/
and /home/hadoop/tmp/dfs/namesecondary/current

while [ 1 ]
do
    ls /home/hadoop/tmp/dfs/name/
    ls -lht /home/hadoop/tmp/dfs/name/current/
    ls /home/hadoop/tmp/dfs/namesecondary/
    ls -lht /home/hadoop/tmp/dfs/namesecondary/current/
    echo ""
    sleep 1s
done
```

observe_hdfs.sh

```
#!/bin/bash
# Shell for observe hdfs

while [ 1 ]
do
    hadoop fs -la /input
    echo ""
    sleep 1s
done
```

observe_error.sh

```
#!/bin/bash
# Shell for observe DataNode error

while [ 1 ]
do
    echo "master"
    tree /home/hadoop/
    echo "slave-1"
    ssh root@cluster-slave-1 "tree /home/hadoop/"
    echo "slave-2"
    ssh root@cluster-slave-2 "tree /home/hadoop/"
    echo "slave-3"
    ssh root@cluster-slave-3 "tree /home/hadoop/"
    echo "hdfs"
```

```
hadoop fs -la /input  
sleep 2s  
echo ""  
done
```

top_record.sh

```
#!/bin/bash  
# Shell for record top  
sleep_sec=$1  
print_times=$2  
log_file=$3  
  
top -d $sleep_sec -n $print_times -b > $log_file
```

NameNode 主备实验结果：

host_1_3_1.log

```
current  
in_use.lock  
total 4.1M  
-rw-r--r-- 1 root root 1.0M Oct 12 07:59  
edits_inprogress_00000000000000000031  
-rw-r--r-- 1 root root 3 Oct 12 07:35 seen_txid  
-rw-r--r-- 1 root root 212 Oct 12 07:35 VERSION  
-rw-r--r-- 1 root root 62 Oct 12 07:35  
fsimage_000000000000000030.md5  
-rw-r--r-- 1 root root 668 Oct 12 07:35 fsimage_00000000000000000030  
-rw-r--r-- 1 root root 1.0M Oct 11 02:45 edits_00000000000000000030-  
00000000000000000030  
-rw-r--r-- 1 root root 62 Oct 11 02:45  
fsimage_000000000000000029.md5  
-rw-r--r-- 1 root root 668 Oct 11 02:45 fsimage_00000000000000000029  
-rw-r--r-- 1 root root 1.0M Oct 10 01:31 edits_00000000000000000029-  
00000000000000000029  
-rw-r--r-- 1 root root 42 Oct 10 01:31 edits_00000000000000000027-  
00000000000000000028  
-rw-r--r-- 1 root root 1.0M Oct 10 01:22 edits_0000000000000000001-  
00000000000000000026  
current  
total 1.1M  
-rw-r--r-- 1 root root 212 Oct 10 01:32 VERSION  
-rw-r--r-- 1 root root 668 Oct 10 01:32 fsimage_00000000000000000028
```

```
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_000000000000000028.md5
-rw-r--r-- 1 root root 42 Oct 10 01:32 edits_000000000000000027-
000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:32 edits_000000000000000001-
000000000000000026
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_000000000000000000.md5
-rw-r--r-- 1 root root 394 Oct 10 01:32 fsimage_00000000000000000000

current
in_use.lock
total 4.1M
-rw-r--r-- 1 root root 1.0M Oct 12 08:13
edits_inprogress_000000000000000031
-rw-r--r-- 1 root root 3 Oct 12 07:35 seen_txid
-rw-r--r-- 1 root root 212 Oct 12 07:35 VERSION
-rw-r--r-- 1 root root 62 Oct 12 07:35
fsimage_000000000000000030.md5
-rw-r--r-- 1 root root 668 Oct 12 07:35 fsimage_0000000000000000030
-rw-r--r-- 1 root root 1.0M Oct 11 02:45 edits_000000000000000030-
000000000000000030
-rw-r--r-- 1 root root 62 Oct 11 02:45
fsimage_000000000000000029.md5
-rw-r--r-- 1 root root 668 Oct 11 02:45 fsimage_000000000000000029
-rw-r--r-- 1 root root 1.0M Oct 10 01:31 edits_000000000000000029-
000000000000000029
-rw-r--r-- 1 root root 42 Oct 10 01:31 edits_000000000000000027-
000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:22 edits_000000000000000001-
000000000000000026
current
total 1.1M
-rw-r--r-- 1 root root 212 Oct 10 01:32 VERSION
-rw-r--r-- 1 root root 668 Oct 10 01:32 fsimage_000000000000000028
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_000000000000000028.md5
-rw-r--r-- 1 root root 42 Oct 10 01:32 edits_000000000000000027-
000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:32 edits_000000000000000001-
000000000000000026
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_000000000000000000.md5
-rw-r--r-- 1 root root 394 Oct 10 01:32 fsimage_00000000000000000000
```

.....

```
current
in_use.lock
total 4.1M
-rw-r--r-- 1 root root 1.0M Oct 12 08:13
 edits_inprogress_00000000000000000031
-rw-r--r-- 1 root root      3 Oct 12 07:35 seen_txid
-rw-r--r-- 1 root root  212 Oct 12 07:35 VERSION
-rw-r--r-- 1 root root     62 Oct 12 07:35
fsimage_00000000000000000030.md5
-rw-r--r-- 1 root root   668 Oct 12 07:35 fsimage_00000000000000000030
-rw-r--r-- 1 root root 1.0M Oct 11 02:45 edits_00000000000000000030-
00000000000000000030
-rw-r--r-- 1 root root     62 Oct 11 02:45
fsimage_00000000000000000029.md5
-rw-r--r-- 1 root root   668 Oct 11 02:45 fsimage_00000000000000000029
-rw-r--r-- 1 root root 1.0M Oct 10 01:31 edits_00000000000000000029-
00000000000000000029
-rw-r--r-- 1 root root     42 Oct 10 01:31 edits_00000000000000000027-
00000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:22 edits_0000000000000000001-
00000000000000000026
current
total 1.1M
-rw-r--r-- 1 root root  212 Oct 10 01:32 VERSION
-rw-r--r-- 1 root root   668 Oct 10 01:32 fsimage_00000000000000000028
-rw-r--r-- 1 root root     62 Oct 10 01:32
fsimage_00000000000000000028.md5
-rw-r--r-- 1 root root     42 Oct 10 01:32 edits_00000000000000000027-
00000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:32 edits_0000000000000000001-
00000000000000000026
-rw-r--r-- 1 root root     62 Oct 10 01:32
fsimage_00000000000000000000.md5
-rw-r--r-- 1 root root   394 Oct 10 01:32 fsimage_00000000000000000000
```

host_1_3_2.log

```
.....
current
in_use.lock
total 4.1M
-rw-r--r-- 1 root root 1.0M Oct 12 08:13
 edits_inprogress_00000000000000000031
-rw-r--r-- 1 root root      3 Oct 12 07:35 seen_txid
-rw-r--r-- 1 root root  212 Oct 12 07:35 VERSION
```

```
-rw-r--r-- 1 root root 62 Oct 12 07:35
fsimage_00000000000000000030.md5
-rw-r--r-- 1 root root 668 Oct 12 07:35 fsimage_00000000000000000030
-rw-r--r-- 1 root root 1.0M Oct 11 02:45 edits_00000000000000000030-
00000000000000000030
-rw-r--r-- 1 root root 62 Oct 11 02:45
fsimage_00000000000000000029.md5
-rw-r--r-- 1 root root 668 Oct 11 02:45 fsimage_00000000000000000029
-rw-r--r-- 1 root root 1.0M Oct 10 01:31 edits_00000000000000000029-
00000000000000000029
-rw-r--r-- 1 root root 42 Oct 10 01:31 edits_00000000000000000027-
00000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:22 edits_0000000000000000001-
00000000000000000026
current
total 1.1M
-rw-r--r-- 1 root root 212 Oct 10 01:32 VERSION
-rw-r--r-- 1 root root 668 Oct 10 01:32 fsimage_00000000000000000028
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_00000000000000000028.md5
-rw-r--r-- 1 root root 42 Oct 10 01:32 edits_00000000000000000027-
00000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:32 edits_0000000000000000001-
00000000000000000026
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_0000000000000000000000000000000000
-rw-r--r-- 1 root root 394 Oct 10 01:32 fsimage_0000000000000000000000000000000000

current
in_use.lock
total 4.1M
-rw-r--r-- 1 root root 1.0M Oct 12 08:18
edits_inprogress_00000000000000000031
-rw-r--r-- 1 root root 3 Oct 12 07:35 seen_txid
-rw-r--r-- 1 root root 212 Oct 12 07:35 VERSION
-rw-r--r-- 1 root root 62 Oct 12 07:35
fsimage_00000000000000000030.md5
-rw-r--r-- 1 root root 668 Oct 12 07:35 fsimage_00000000000000000030
-rw-r--r-- 1 root root 1.0M Oct 11 02:45 edits_00000000000000000030-
00000000000000000030
-rw-r--r-- 1 root root 62 Oct 11 02:45
fsimage_00000000000000000029.md5
-rw-r--r-- 1 root root 668 Oct 11 02:45 fsimage_00000000000000000029
-rw-r--r-- 1 root root 1.0M Oct 10 01:31 edits_00000000000000000029-
00000000000000000029
-rw-r--r-- 1 root root 42 Oct 10 01:31 edits_00000000000000000027-
00000000000000000028
```

```

-rw-r--r-- 1 root root 1.0M Oct 10 01:22 edits_000000000000000000001-
0000000000000000026
current
total 1.1M
-rw-r--r-- 1 root root 212 Oct 10 01:32 VERSION
-rw-r--r-- 1 root root 668 Oct 10 01:32 fsimage_00000000000000000028
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_0000000000000000028.md5
-rw-r--r-- 1 root root 42 Oct 10 01:32 edits_00000000000000000027-
0000000000000000028
-rw-r--r-- 1 root root 1.0M Oct 10 01:32 edits_000000000000000000001-
0000000000000000026
-rw-r--r-- 1 root root 62 Oct 10 01:32
fsimage_000000000000000000000000000000000000000000000000000000000000000
......


```

hdfs_1_3_1.log

drwxr-xr-x	- root staff	0 2020-10-12 08:13	/input
-rw-r--r--	3 root staff	0 2020-10-12 08:13	/input/hadoop- dis.tar._COPYING_
drwx-----	- root staff	0 2020-10-12 07:46	/user
drwx-----	- root staff	0 2020-10-12 07:46	/user/root
drwx-----	- root staff	0 2020-10-12 07:46	/user/root/.Trash
drwx-----	- root staff	0 2020-10-12 07:46	
	/user/root/.Trash/Current		
drwx-----	- root staff	0 2020-10-12 07:59	
	/user/root/.Trash/Current/input		
-rw-r--r--	3 root staff 915056640	2020-10-10 01:22	
	/user/root/.Trash/Current/input/hadoop-dis.tar		
-rw-r--r--	3 root staff 915056640	2020-10-12 07:51	
	/user/root/.Trash/Current/input/hadoop-dis.tar1602489554685		
drwxr-xr-x	- root staff	0 2020-10-12 08:13	/input
-rw-r--r--	3 root staff 915056640	2020-10-12 08:13	/input/hadoop- dis.tar
drwx-----	- root staff	0 2020-10-12 07:46	/user
drwx-----	- root staff	0 2020-10-12 07:46	/user/root
drwx-----	- root staff	0 2020-10-12 07:46	/user/root/.Trash
drwx-----	- root staff	0 2020-10-12 07:46	
	/user/root/.Trash/Current		
drwx-----	- root staff	0 2020-10-12 07:59	
	/user/root/.Trash/Current/input		

```
-rw-r--r--  3 root staff  915056640 2020-10-10 01:22  
/user/root/.Trash/Current/input/hadoop-dis.tar  
-rw-r--r--  3 root staff  915056640 2020-10-12 07:51  
/user/root/.Trash/Current/input/hadoop-dis.tar1602489554685  
  
drwxr-xr-x  - root staff          0 2020-10-12 08:13 /input  
-rw-r--r--  3 root staff  915056640 2020-10-12 08:13 /input/hadoop-  
dis.tar  
drwx-----  - root staff          0 2020-10-12 07:46 /user  
drwx-----  - root staff          0 2020-10-12 07:46 /user/root  
drwx-----  - root staff          0 2020-10-12 07:46 /user/root/.Trash  
drwx-----  - root staff          0 2020-10-12 07:46  
/user/root/.Trash/Current  
drwx-----  - root staff          0 2020-10-12 07:59  
/user/root/.Trash/Current/input  
-rw-r--r--  3 root staff  915056640 2020-10-10 01:22  
/user/root/.Trash/Current/input/hadoop-dis.tar  
-rw-r--r--  3 root staff  915056640 2020-10-12 07:51  
/user/root/.Trash/Current/input/hadoop-dis.tar1602489554685
```

hdfs_1_3_2.log

```
drwxr-xr-x  - root staff          0 2020-10-12 08:13 /input  
-rw-r--r--  3 root staff  915056640 2020-10-12 08:13 /input/hadoop-  
dis.tar  
drwx-----  - root staff          0 2020-10-12 07:46 /user  
drwx-----  - root staff          0 2020-10-12 07:46 /user/root  
drwx-----  - root staff          0 2020-10-12 07:46 /user/root/.Trash  
drwx-----  - root staff          0 2020-10-12 07:46  
/user/root/.Trash/Current  
drwx-----  - root staff          0 2020-10-12 07:59  
/user/root/.Trash/Current/input  
-rw-r--r--  3 root staff  915056640 2020-10-10 01:22  
/user/root/.Trash/Current/input/hadoop-dis.tar  
-rw-r--r--  3 root staff  915056640 2020-10-12 07:51  
/user/root/.Trash/Current/input/hadoop-dis.tar1602489554685  
  
drwxr-xr-x  - root staff          0 2020-10-12 08:18 /input  
drwx-----  - root staff          0 2020-10-12 07:46 /user  
drwx-----  - root staff          0 2020-10-12 07:46 /user/root  
drwx-----  - root staff          0 2020-10-12 07:46 /user/root/.Trash  
drwx-----  - root staff          0 2020-10-12 07:46  
/user/root/.Trash/Current  
drwx-----  - root staff          0 2020-10-12 08:18  
/user/root/.Trash/Current/input
```

```
-rw-r--r--  3 root staff  915056640 2020-10-10 01:22
/usr/root/.Trash/Current/input/hadoop-dis.tar
-rw-r--r--  3 root staff  915056640 2020-10-12 07:51
/usr/root/.Trash/Current/input/hadoop-dis.tar1602489554685
-rw-r--r--  3 root staff  915056640 2020-10-12 08:13
/usr/root/.Trash/Current/input/hadoop-dis.tar1602490723925
```

```
drwxr-xr-x  - root staff      0 2020-10-12 08:18 /input
drwx-----  - root staff      0 2020-10-12 07:46 /user
drwx-----  - root staff      0 2020-10-12 07:46 /user/root
drwx-----  - root staff      0 2020-10-12 07:46 /user/root/.Trash
drwx-----  - root staff      0 2020-10-12 07:46
/usr/root/.Trash/Current
drwx-----  - root staff      0 2020-10-12 08:18
/usr/root/.Trash/Current/input
-rw-r--r--  3 root staff  915056640 2020-10-10 01:22
/usr/root/.Trash/Current/input/hadoop-dis.tar
-rw-r--r--  3 root staff  915056640 2020-10-12 07:51
/usr/root/.Trash/Current/input/hadoop-dis.tar1602489554685
-rw-r--r--  3 root staff  915056640 2020-10-12 08:13
/usr/root/.Trash/Current/input/hadoop-dis.tar1602490723925
```

```
drwxr-xr-x  - root staff      0 2020-10-12 08:18 /input
drwx-----  - root staff      0 2020-10-12 07:46 /user
drwx-----  - root staff      0 2020-10-12 07:46 /user/root
drwx-----  - root staff      0 2020-10-12 07:46 /user/root/.Trash
drwx-----  - root staff      0 2020-10-12 07:46
/usr/root/.Trash/Current
drwx-----  - root staff      0 2020-10-12 08:18
/usr/root/.Trash/Current/input
-rw-r--r--  3 root staff  915056640 2020-10-10 01:22
/usr/root/.Trash/Current/input/hadoop-dis.tar
-rw-r--r--  3 root staff  915056640 2020-10-12 07:51
/usr/root/.Trash/Current/input/hadoop-dis.tar1602489554685
-rw-r--r--  3 root staff  915056640 2020-10-12 08:13
/usr/root/.Trash/Current/input/hadoop-dis.tar1602490723925
```

```
drwxr-xr-x  - root staff      0 2020-10-12 08:18 /input
drwx-----  - root staff      0 2020-10-12 07:46 /user
drwx-----  - root staff      0 2020-10-12 07:46 /user/root
drwx-----  - root staff      0 2020-10-12 07:46 /user/root/.Trash
drwx-----  - root staff      0 2020-10-12 07:46
/usr/root/.Trash/Current
drwx-----  - root staff      0 2020-10-12 08:18
/usr/root/.Trash/Current/input
```

```
-rw-r--r-- 3 root staff 915056640 2020-10-10 01:22
/usr/root/.Trash/Current/input/hadoop-dis.tar
-rw-r--r-- 3 root staff 915056640 2020-10-12 07:51
/usr/root/.Trash/Current/input/hadoop-dis.tar1602489554685
-rw-r--r-- 3 root staff 915056640 2020-10-12 08:13
/usr/root/.Trash/Current/input/hadoop-dis.tar1602490723925
```

hdfs_1_3.log

```
drwxr-xr-x - root staff 0 2020-10-12 08:18 /input
drwx----- - root staff 0 2020-10-12 07:46 /user
drwx----- - root staff 0 2020-10-12 07:46 /user/root
drwx----- - root staff 0 2020-10-12 07:46 /user/root/.Trash
drwx----- - root staff 0 2020-10-12 07:46
/usr/root/.Trash/Current
drwx----- - root staff 0 2020-10-12 08:18
/usr/root/.Trash/Current/input
-rw-r--r-- 3 root staff 915056640 2020-10-10 01:22
/usr/root/.Trash/Current/input/hadoop-dis.tar
-rw-r--r-- 3 root staff 915056640 2020-10-12 07:51
/usr/root/.Trash/Current/input/hadoop-dis.tar1602489554685
-rw-r--r-- 3 root staff 915056640 2020-10-12 08:13
/usr/root/.Trash/Current/input/hadoop-dis.tar1602490723925
```

```
ls: Call From cluster-master/10.0.0.2 to cluster-master:9000 failed on
connection exception: java.net.ConnectException: Connection refused;
For more details see: http://wiki.apache.org/hadoop/ConnectionRefused
```

hdfs_1_3_4.log

```
ls: Call From cluster-master/10.0.0.2 to cluster-master:9000 failed on
connection exception: java.net.ConnectException: Connection refused;
For more details see: http://wiki.apache.org/hadoop/ConnectionRefused
```

DataNode 故障恢复实验结果：

由于这部分实验结果行数太多了，因此不在这里放出来，具体可以到 github 上查看：[datanode_log](#)

Hadoop 参数影响实验结果：

top_1M.log

top_100M.log

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+
-----	------	----	----	------	-----	-----	---	------	------	-------

top_500M.log

自主设计黑盒实验数据

top_1_6_4.log

	PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+
sshd	44	root	20	0	78528	2028	1084	S	0.0	0.1	0:00.03
bash	71	root	20	0	12148	2644	2016	S	0.0	0.1	0:00.16
bash	2635	root	20	0	12148	2240	1648	S	0.0	0.1	0:00.07
top_record.sh	9231	root	20	0	11884	2724	2496	S	0.0	0.1	0:00.00
COMMAND											
java	9233	root	20	0	2681756	338112	24300	S	12.5	8.4	0:09.42
java	8169	root	20	0	2747908	225724	25244	S	11.0	5.6	0:12.08
java	8596	root	20	0	2907960	322124	25456	S	1.5	8.0	0:11.43
java	8023	root	20	0	2759388	294212	25900	S	0.5	7.3	0:09.42
java	8340	root	20	0	2700644	142196	25240	S	0.5	3.5	0:04.65
java	8772	root	20	0	2781144	228584	26252	S	0.5	5.7	0:08.89
systemd	1	root	20	0	94400	2140	0	S	0.0	0.1	0:00.16
systemd-journal	26	root	20	0	93912	3896	2904	S	0.0	0.1	0:00.39
systemd-udevd	28	root	20	0	88500	1652	212	S	0.0	0.0	0:00.58
dbus-daemon	43	dbus	20	0	53976	556	0	S	0.0	0.0	0:00.11
sshd	44	root	20	0	78528	2004	1060	S	0.0	0.0	0:00.03
bash	71	root	20	0	12148	2620	1992	S	0.0	0.1	0:00.16
bash	2635	root	20	0	12148	2216	1624	S	0.0	0.1	0:00.07
top_record.sh	9231	root	20	0	11884	2700	2472	S	0.0	0.1	0:00.00
top	9232	root	20	0	48920	3712	3176	R	0.0	0.1	0:00.02

top_1_6_7.log

System Load and Process Activity								
Process Details		System Metrics		CPU Usage		Memory Usage		
User	Process Name	CPU %	Memory (kB)	Time	State	Memory %	Swap %	Virtual / Real
4439	root	20	0	2674204	175464	25312	S	88.5
java		20	0	2738804	191300	15856	S	6.5
2999	root	20	0	2784436	334484	16016	S	1.0
java		20	0	2917044	318912	15732	S	1.0
2853	root	20	0	2786388	236812	15660	S	5.9
java		20	0	94400	3104	964	S	0.0
3456	root	20	0	93912	2788	1816	S	0.0
java		20	0	88500	2060	620	S	0.0
3602	root	20	0	53976	416	0	S	0.0
systemd		20	0	78528	1568	624	S	0.0
26	root	20	0	12148	2388	1780	S	0.0
systemd-journal		20	0	12148	2244	1652	S	0.0
28	root	20	0	2700648	138804	15648	S	3.4
systemd-udevd		20	0	11884	2768	2540	S	0.0
43	dbus	20	0	48920	3656	3124	R	0.1
dbus-daemon		20	0	48920	3656	3124	R	0.0
44	root	20	0	48920	3656	3124	R	0.0
sshd		20	0	48920	3656	3124	R	0.0
71	root	20	0	48920	3656	3124	R	0.0
bash		20	0	48920	3656	3124	R	0.0
2635	root	20	0	48920	3656	3124	R	0.0
bash		20	0	48920	3656	3124	R	0.0
3171	root	20	0	48920	3656	3124	R	0.0
java		20	0	48920	3656	3124	R	0.0
4437	root	20	0	48920	3656	3124	R	0.0
top_record.sh		20	0	48920	3656	3124	R	0.0
4438	root	20	0	48920	3656	3124	R	0.0
top		20	0	48920	3656	3124	R	0.0


```
4438 root      20  0  48920  3656  3124 R  0.0  0.1  0:00.02
top
```

top_1_6_9.log

