实验题目：HDFS文件系统的基本操作与性能分析

实验目的：

1. 掌握HDFS的基本操作，包括文件上传、下载、删除等。
2. 理解HDFS的架构及其工作原理。
3. 进行HDFS性能测试，分析其性能瓶颈。

实验环境：

1. 安装前的准备工作

- 安装Java HDFS依赖于Java运行环境，需要安装Java Development Kit (JDK)。

```
sudo apt update
sudo apt install default-jdk -y
```

验证安装 `java -version`

• 下载Hadoop 前往Hadoop官方网站下载最新版本的Hadoop二进制文件：

```
wget https://downloads.apache.org/hadoop/common/hadoop-3.4.0/hadoop-3.4.0.tar.gz
```

解压下载的文件：

```
tar -xzvf hadoop-3.4.0.tar.gz
sudo mv hadoop-3.4.0 /usr/local/hadoop
```

- 配置环境变量 编辑~/.bashrc文件，添加Hadoop相关的环境变量

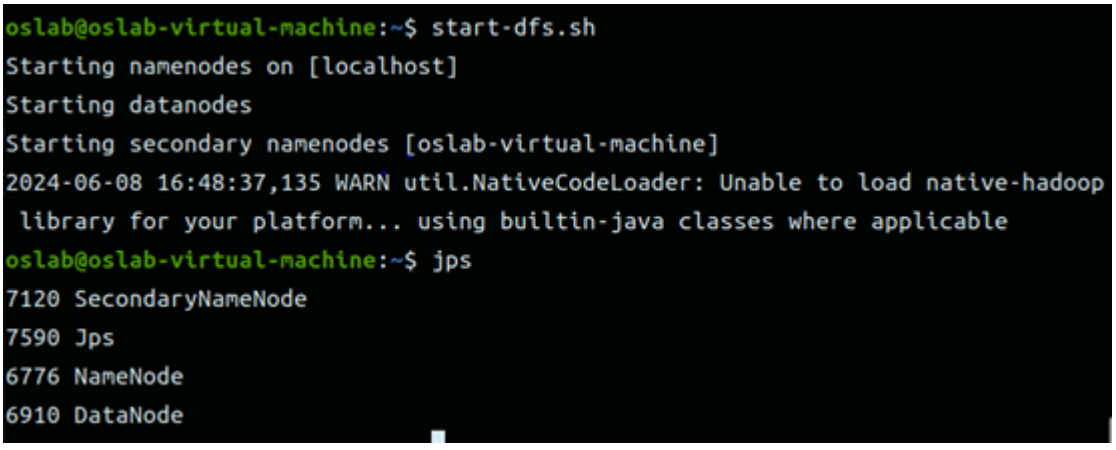
```
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
```

- 应用配置

```
source ~/.bashrc
```

```
添加sshserver
```

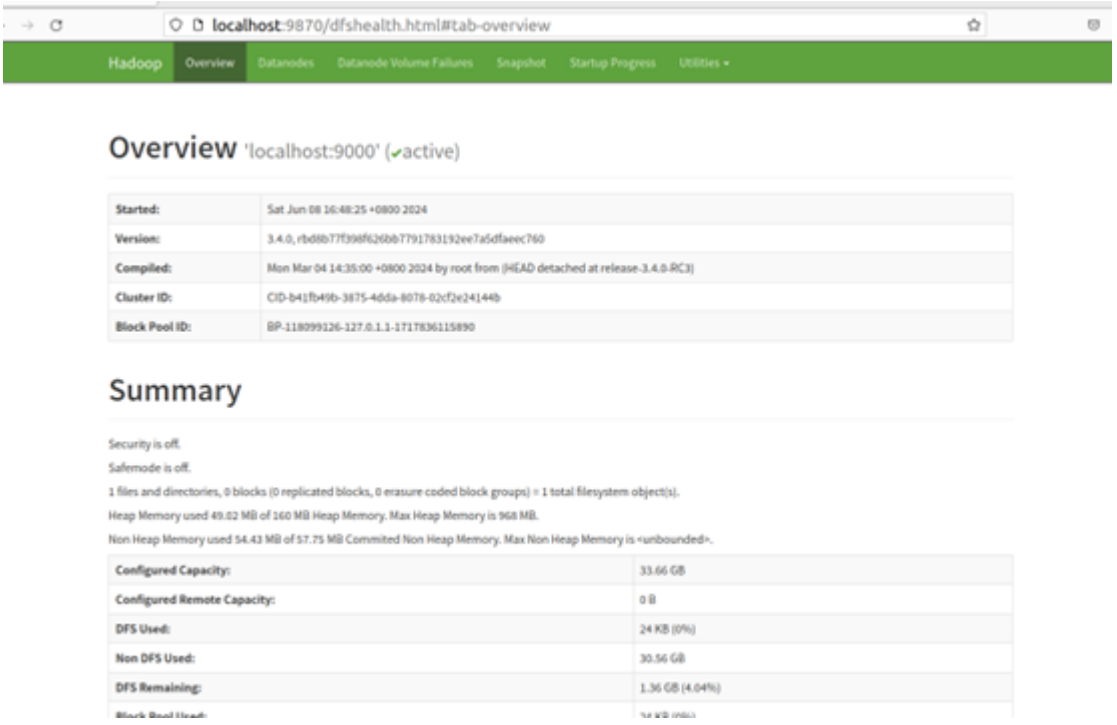
- 启动



- 检查 NameNode Web 界面： 打开浏览器并访问 NameNode 的 Web 界面。默认地址通常是：

> http://localhost:9870

- 在这里你可以看到 HDFS 的状态，包括 DataNode 的数量、容量、使用情况等。



学习HDFS系统架构

> Hadoop分布式文件系统（HDFS）是Apache Hadoop生态系统的关键组件之一，旨在提供高可靠性、高吞吐量的存储服务

架构设计

- **NameNode** HDFS的关键组件之一，负责管理文件系统的命名空间、文件元数据以及块的映射。它维护了文件系统树结构，并跟踪每个文件的块位置和副本信息。
- **DataNode** 集群中实际存储数据的节点。它们负责存储和检索数据块，并根据NameNode的指示进行数据块的复制、移动和删除。
- Secondary NameNode 负责定期合并NameNode的编辑日志（edits log）和文件系统镜像（fsimage），以防止NameNode的元数据丢失。

数据处理流程

> 客户端向NameNode请求文件写入操作，NameNode返回可用的DataNode列表。客户端直接与这些DataNode建立连接，将数据分割成块并写入这些DataNode，同时每个块都会在集群中复制多个副本。

客户端向NameNode请求文件读取操作，NameNode返回文件的块位置信息。客户端直接与存储有所需块副本的DataNode建立连接，并从它们读取数据。

- 工作机制： 当客户端向 HDFS 写入文件时，数据被分成固定大小的块，并分配给不同的 DataNode 存储。NameNode 记录每个文件的元数据信息和数据块的位置信息。 当客户端需要读取文件时，它首先请求 NameNode 获取文件的元数据信息和数据块的位置信息，然后直接与对应的 DataNode 进行通信获取数据块。
- 文件存储机制 在 HDFS 中，文件被分割成固定大小的块（block），默认情况下是 128MB。这些块以相同的大小分布在不同的 DataNode 上，以提高数据的可靠性和可用性。 查看文件的块分布情况

特性和优势

HDFS通过数据冗余和自动故障转移确保了高可靠性和数据持久性。：HDFS设计用于支持大规模数据集的高吞吐量读写操作，适用于批量数据处理场景。通过添加更多的节点来扩展存储容量和处理能力，实现了良好的横向扩展性。可以与Hadoop生态系统中的其他组件（如MapReduce、Hive、Spark等）无缝集成，支持大规模数据处理和分析。

实验步骤

HDFS基本操作

- 文件上传与下载
  上传一个本地文件到HDFS的指定目录。
  从HDFS下载文件到本地。![1717839865600](image/HDFS实验/1717839865600.png)
- 文件删除
- 删除HDFS中的指定文件或目录。

```
hdfs dfs -rm /test/test.txt
hdfs dfs -rm -r /test
```

文件查看
查看HDFS目录中的文件列表和文件内容。

```
hdfs dfs -ls /test
hdfs dfs -cat /test/test.txt
```

```
oslab@oslab-virtual-machine:~$ hdfs dfs -ls /test
hdfs dfs -cat /test/test.txt
2024-06-08 18:03:01,784 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 oslab supergroup          0 2024-06-08 17:14 /test/test.txt
2024-06-08 18:03:06,292 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable
```

```
hdfs fsck /test -files -blocks -locations
```

这个命令将会列出指定文件的元数据信息，包括文件块的数量、每个块的大小以及存储位置等信息。！[1717841074496](image/HDFS实验/1717841074496.png)

HDFS性能测试

- 大文件读写性能测试 上传一个大文件（如10GB）到HDFS，并记录时间。 从HDFS读取该文件，并记录时间。 使用time命令记录时间：

```
time hdfs dfs -put ./large.txt /test/
time hdfs dfs -get /test/large.txt local_largefile_copy.txt
```

```
oslab@oslab-virtual-machine:~/Desktop$ time hdfs dfs -put ./large.txt /test/
2024-06-08 18:09:02,921 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable

real    0m19.333s
user    0m14.415s
sys     0m4.577s
```

```
oslab@oslab-virtual-machine:~/Desktop$ time hdfs dfs -get /test/large.txt local_
largefile_copy.txt
2024-06-08 18:10:11,372 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable

real    0m8.756s
user    0m15.305s
sys     0m3.845s
oslab@oslab-virtual-machine:~/Desktop$
```

- 小文件读写性能测试

  未做

性能提升方法构想

- 增加 DataNode 数量 通过增加 DataNode 节点来提高存储容量和并行处理能力，从而提高整体性能。
- 调整块大小：根据文件大小和访问模式调整块大小，以减少存储空间浪费和减轻元数据管理负载。