

实验1：Linux、AI助手与HDFS

4学时 · 起点镜像 L01

今天的成果：一个能存入HDFS、再取回核对的文件

实验任务与课堂安排

阶段	时间	检查结果
连接与文件	35分钟	student身份、个人文件
AI与配置	35分钟	读取配置与确认访问地址
HDFS存取	55分钟	上传、读取、下载一致
修改与排错	30分钟	个人文件、路径修正
核对	25分钟	Word报告填写与核对

今天需要完成什么

分清Windows宿主机与Linux虚拟机

通过SSH确认自己的用户和位置

查看一份HDFS配置，启动两个核心进程

上传、读取、下载同一个文件

将实际运行结果与模板已有书面解释填写到Word报告。

五个实验的学习顺序

实验	主要任务
1	Linux连接与HDFS文件操作
2	Python MapReduce
3	PySpark数据分析
4	ZooKeeper协调
5	实时统计与恢复

每个实验配有独立起点，按本周编号开始。

Windows、Linux与SSH

Windows运行虚拟化软件、WorkBuddy和浏览器

Linux运行Hadoop、Spark和你的Python程序

SSH把Windows的终端连接到Linux

`C:\Users\...` 是Windows路径； `/home/student/...` 是Linux路径。

虚拟机镜像与检查点

镜像保存操作系统、软件、配置和输入数据

L01提供Hadoop与待完成的文件操作练习

每个实验镜像独立启动，按实验编号领取

个人代码放在 `~/work`，Word报告保存在Windows

本周入口：**L01镜像、lab01指导书、lab01工作目录**

虚拟磁盘与U盘容量

参数	含义
6GB内存	虚拟机运行时占用宿主机内存
40GB动态磁盘	磁盘文件可增长的上限
发布包大小	下载时的压缩归档体积
32GB exFAT U盘	携带当前关机副本和作业

运行位置采用电脑本地磁盘。

导入本次镜像

VMware打开解压后的VMX；VirtualBox导入OVA

分配4个CPU、6GB内存，使用NAT网络

先复制到电脑本地磁盘，再启动

用户名为 `student`，密码留空

32GB U盘携带当前镜像和作业。

确认身份和位置

```
whoami  
hostname  
pwd  
hostname -I
```

预期用户是student；记录自己的虚拟机IP。

NAT地址与连接端口

平台	Windows连接目标	Linux服务
VMware NAT	客体实际IP、22端口	SSH
VirtualBox NAT	127.0.0.1:22220	转发到客体22

127.0.0.1 表示当前执行命令的这台机器。

先在虚拟机控制台运行 `hostname -I` 记录实际地址。

空密码账号与非交互连接

账号： `student`，密码留空。控制台提示密码时直接回车。

```
# 先手动连接一次并核对主机指纹，再验证：
ssh -o BatchMode=yes student@虚拟机IP 'whoami;
hostname '
# VirtualBox:
ssh -o BatchMode=yes -p 22220 student@127.0.0.1
whoami
```

预期直接返回 `student`，WorkBuddy采用同一连接命令。

从Windows连接

```
# VMware: 替换为实际客体IP  
ssh student@192.168.80.128  
# VirtualBox: 课前配置NAT端口转发  
ssh -p 22220 student@127.0.0.1
```

先手动连接成功，再把相同命令交给AI。

终端提示符与命令结构

```
student@dc-lab:~$ ls -lah /home/student/work
```

`student@dc-lab`：用户和主机

`~`：当前目录为家目录

`ls`：程序，`-lah`：选项，末尾路径：参数

输入命令时从 `ls` 开始，提示符由终端显示

绝对路径、相对路径与家目录

写法	在本次实验中的位置
<code>/home/student/work/lab01/hello.txt</code>	从根目录开始的绝对路径
<code>hello.txt</code>	当前目录中的文件
<code>~/work/lab01</code>	student家目录下的实验目录
<code>..</code>	当前目录的上一级

先 `pwd` 确认位置，再用 `ls` 确认文件名。

找到自己的实验目录

```
cd ~/work/lab01  
ls -lah  
cat hello.txt  
mkdir -p evidence
```

~ 代表当前用户的家目录。

查看与编辑文件

```
cp hello.txt my-message.txt  
nano my-message.txt  
cat my-message.txt
```

nano: Ctrl+O保存, 回车确认, Ctrl+X退出。

文件、目录与权限

```
ls -l hello.txt
```

```
# 例如: -rw-r--r-- 1 student student ... hello.txt
```

第一个字符区分普通文件 `-` 与目录 `d`

`r` 可读、`w` 可写、`x` 可执行或进入目录

`~/work` 保存个人修改, `/opt/course` 保存原始材料

Python脚本可用 `python 文件.py` 运行

WorkBuddy的一次完整操作

SSH执行 `whoami` 与 `pwd`，确认环境

读取 `/opt/course/handbook/lab01.md`

复制 `hello.txt` 为 `my-message.txt`

修改个人文件并执行 `cat my-message.txt`

将命令、修改内容和输出记录到报告

提示词包含连接命令、目标、路径与验证方法。

让AI先检查实际环境

给WorkBuddy明确的信息：

已验证的SSH命令和student账户

工作目录 `/home/student/work/lab01`

先读取配置，解释目的，再做最小修改

每步运行后用输出验证

诊断工具与原生命令

```
labctl doctor  
java -version  
python --version
```

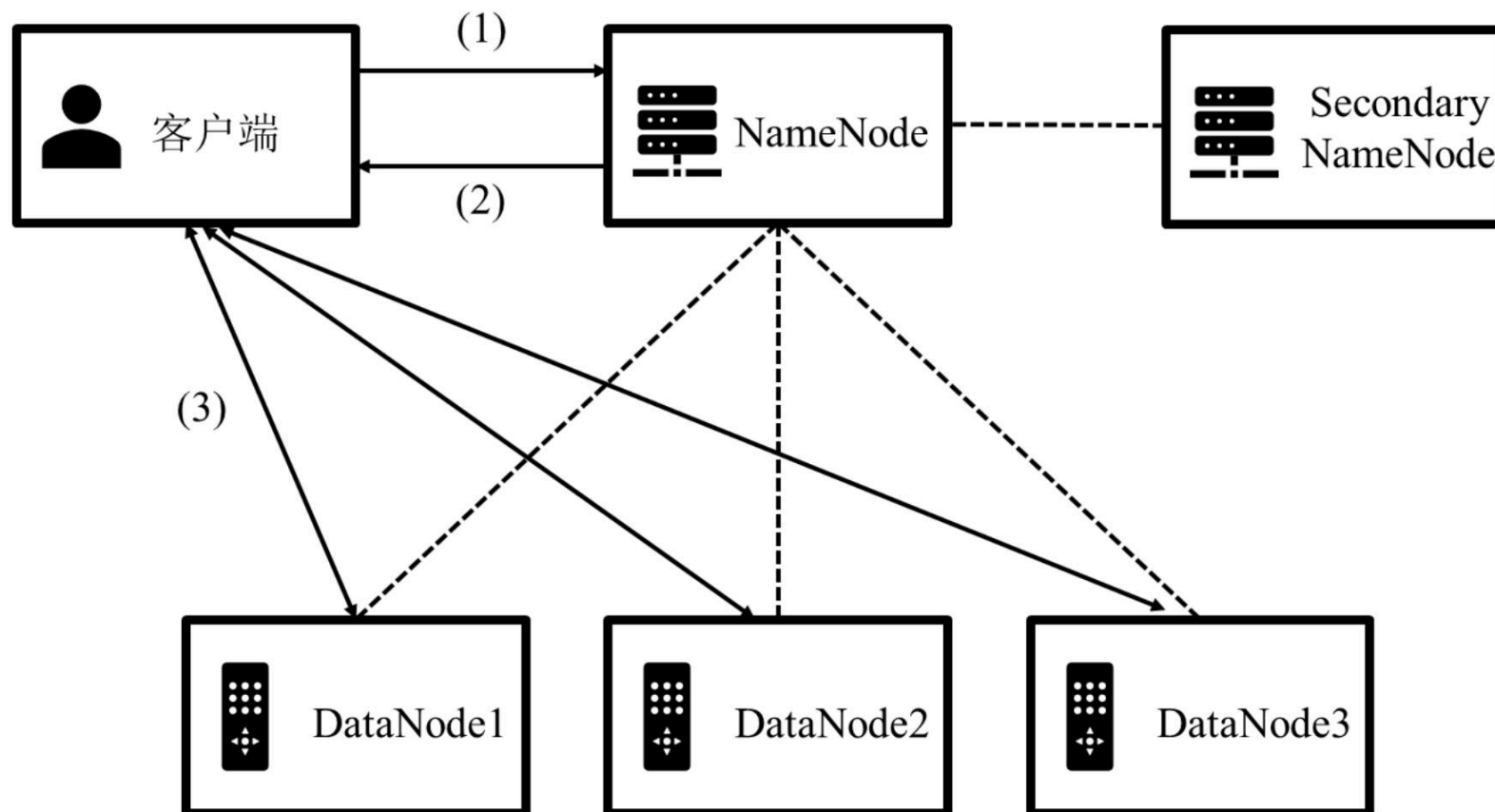
labctl 检查软件与配置，原生命令展示具体版本。

HDFS的客户端、元数据与数据块

客户端向NameNode查询目录和块位置

客户端与DataNode传输文件内容

本实验在同一虚拟机观察不同进程的协作



文件分块与副本数

文件按块存放，NameNode记录文件与块的对应关系

DataNode保存块内容，向NameNode报告存活与块信息

`dfs.replication=1`：新建文件默认期望每块一个副本

`fsck` 显示文件块的实际副本数和所在节点

本实验用小文件观察块位置与副本配置。

HDFS的两个核心角色

角色	本次观察
NameNode	文件名、目录与数据块位置等元数据
DataNode	实际保存数据块并报告状态

本实验采用一个DataNode，副本数配置为1。

两个文件系统

```
ls ~/work/lab01  
hdfs dfs -ls /
```

第一个查看Linux本地文件；第二个查看HDFS。

本地文件通过 `-put` 上传到HDFS。

XML配置项的结构

```
<property>  
  <name>dfs.replication</name>  
  <value>1</value>  
</property>
```

`name` 指定参数名， `value` 指定参数值。

读取配置项，随后用实际上传的文件核对块位置。

默认文件系统地址

```
hdfs://localhost:9000
```

部分	含义
<code>hdfs://</code>	HDFS访问协议
<code>localhost</code>	Linux客体自身
<code>9000</code>	NameNode RPC端口

浏览器的HDFS管理页面采用另一个端口 `9870` 。

查看配置

```
cat ~/conf/hadoop/core-site.xml  
hdfs getconf -confKey fs.defaultFS  
hdfs getconf -confKey dfs.replication
```

`fs.defaultFS`：默认访问哪个HDFS

`dfs.replication`：期望的数据副本数，本环境为1

启动并确认可用

```
labctl start lab01  
jps -l  
hdfs dfsadmin -report
```

确认NameNode、DataNode, 以及一个存活DataNode。

进程、端口与文件读写检查

检查层次	命令	观察对象
进程	<code>jps -l</code>	NameNode、DataNode
端口	<code>ss -ltn</code>	服务监听地址
节点	<code>hdfs dfsadmin -report</code>	Live datanodes为1
内容	<code>hdfs dfs -cat 文件</code>	实际返回的数据

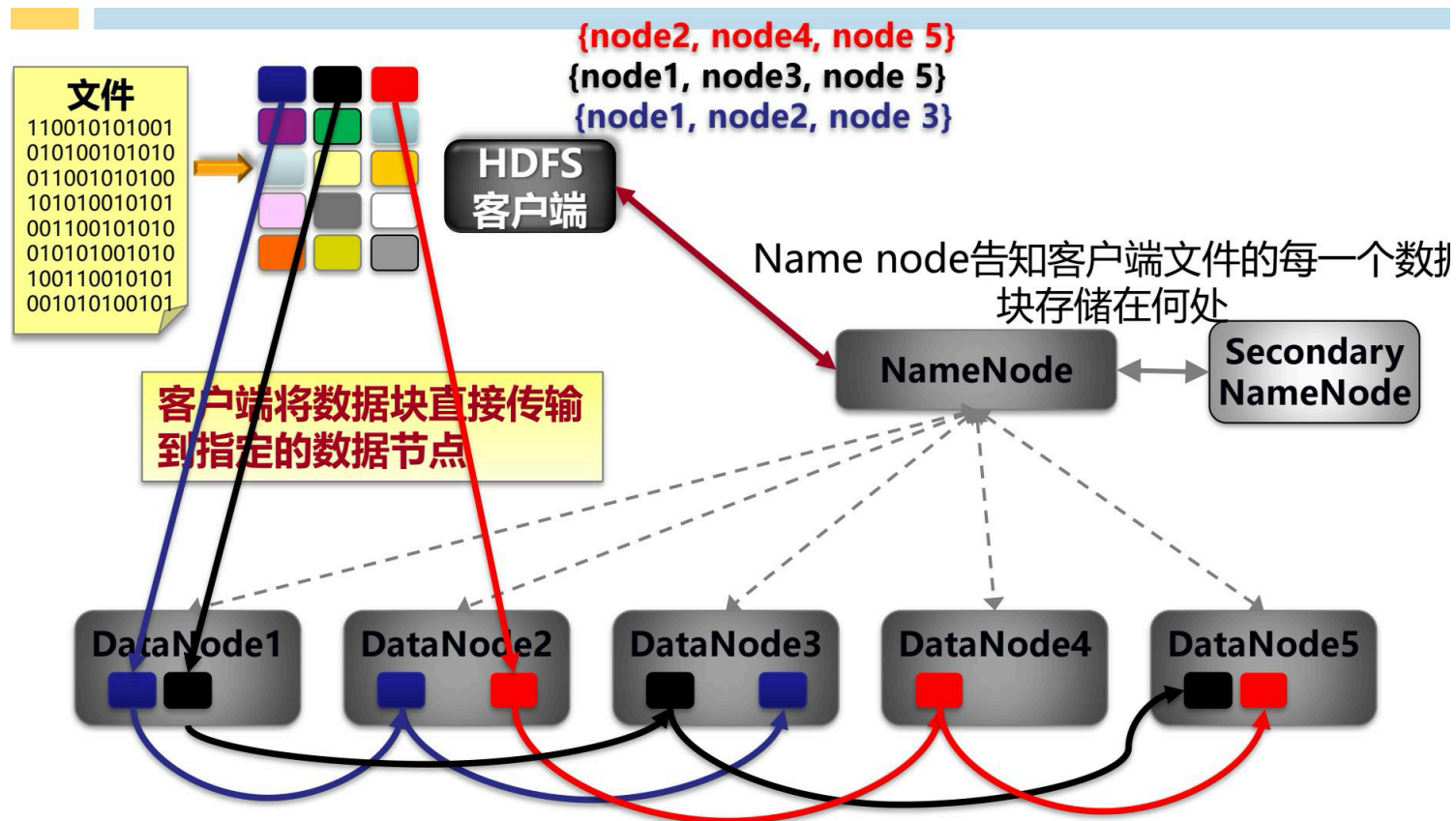
按层检查，可以缩小故障范围。

上传文件的执行过程

客户端向NameNode申请创建文件并获得块位置

客户端向DataNode传输内容

完成写入后，通过读取命令核对文件



HDFS命令中的参数与方向

命令	参数方向
<code>-mkdir -p 路径</code>	在HDFS中创建目录
<code>-put 本地文件 HDFS路径</code>	本地上传到HDFS
<code>-cat HDFS文件</code>	将HDFS内容输出到终端
<code>-get HDFS文件 本地文件</code>	HDFS下载到本地
<code>-ls HDFS目录</code>	列出HDFS目录内容
<code>-f</code> 允许覆盖命令明确指定的目标文件。	

把文件存入HDFS

```
hdfs dfs -mkdir -p /course/lab01
hdfs dfs -put -f hello.txt /course/lab01/hello.txt
hdfs dfs -ls /course/lab01
hdfs dfs -cat /course/lab01/hello.txt
```

读到 `hello distributed computing` 才算核对内容。

下载后比较

```
hdfs dfs -get -f /course/lab01/hello.txt  
downloaded.txt  
diff hello.txt downloaded.txt  
labctl check lab01
```

`diff` 没有输出：两个文件内容一致。

下载结果的内容核对

```
wc -c hello.txt downloaded.txt  
sha256sum hello.txt downloaded.txt  
diff hello.txt downloaded.txt
```

字节数便于快速检查体积

相同SHA-256用于核对内容

`diff` 相同时返回0，并保持空输出

文件块的实际保存位置

```
hdfs fsck /course/lab01/hello.txt -files -blocks -locations
```

检查项	本实验预期
数据块数	1
Live_repl	1
位置列表	1个DataNode地址
状态	HEALTHY

将节点地址与 `hdfs dfsadmin -report` 中的 `Name` 对照。

SSH网页隧道

```
ssh -L 19870:127.0.0.1:9870 student@虚拟机IP
```

左侧19870: Windows本机浏览器访问的端口

右侧9870: Linux客体中的HDFS网页端口

浏览器打开 <http://127.0.0.1:19870>

SSH窗口保持开启, HDFS服务保持运行

路径错误的诊断记录

记录项	示例
目标	读取刚上传的文件
失败路径	<code>/course/lab01/not-exist.txt</code>
诊断命令	<code>hdfs dfs -ls /course/lab01</code>
发现	实际文件名为 <code>hello.txt</code>
修复	使用正确路径重新读取

报告保留一段完整的“现象、诊断、修复、验证”。

一次可控的路径错误

```
hdfs dfs -cat /course/lab01/not-exist.txt  
hdfs dfs -ls /course/lab01
```

先从错误里找路径，再列目录核对。

核对目录内容后，使用正确文件名重新读取。

个人修改与解释

上传自己编辑的my-message.txt，再下载核对

解释本地路径与HDFS路径

解释NameNode与DataNode的分工

指出一条验证AI修改是否生效的命令

Word报告填写位置

第1步：确认环境

第2步：确认HDFS访问地址与默认副本数

第3步：启动和观察HDFS

第4步：上传读取和下载

第5步：个人文本与路径修复

按模板栏目粘贴命令、实际输出及已有书面解释。

报告截图核对

图1 HDFS节点状态（报告3 启动和观察HDFS）

图2 文件往返核对（报告4 上传读取和下载）

图3 文件块实际保存位置（报告4 上传读取和下载）

截图内容与图号按Word模板填写，直接插入对应位置。

提交Word报告

完成各步骤实际结果、规定截图和已有书面栏目

AI使用情况：简短说明或一张截图；未使用填“未使用”

保存后重新打开，核对文本和截图清晰可读

正式提交一份填写完成的Word报告。