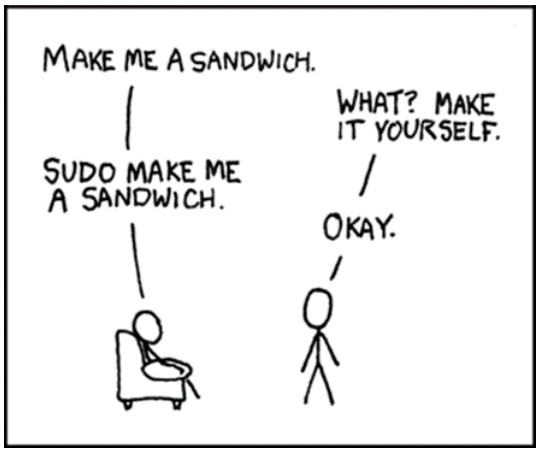
实验一 Linux基础实验



一、实验目标

- 通过对常用命令`cd、ls、cp、mv、rm、mkdir`等文件命令的操作，掌握Linux操作系统中文件命令的用法。（2 分）
- 了解集群中多主机数据传输、任务处理。（3 分）

二、实验任务与要求

- 单机Linux实验
  - 使用`ssh`指令登录远程服务器，并设置免密登录。
  - 掌握`pwd、mkdir、cd`命令的操作，要求能建立目录、进入与退出目录。
  - 掌握`cp、vim、ls、mv、rm`命令的操作，要求能拷贝文件、新建文件、查看文件、文件重命名、删除文件等操作。
  - 掌握`cat、head、scp、awk、grep`等文本处理命令的使用，对文本数据进行查看、过滤、统计等操作。
  - 了解Linux中阻塞与非阻塞的概念，测试两种情况指令执行的情况。
- 多机协同实验
  - 配置集群中**多主机**的免密登录，掌握使用`scp`在不同主机之间传输文件。
  - 在多节点（即多个Linux主机上）上完成任务，对比单节点和多节点的任务处理。
- 服务器介绍
  - 服务器集群由四台主机组成，分别为thumm01，thumm03-05，其中thumm01可由本地ssh连接（需连接网络tsinghua或使用VPN），登录thumm01后，使用命令`ssh thumm0x`可以跳转至其他主机。服务器资源有限，请勿用于课程无关任务。
    <!--
  - 服务器集群由五台主机组成，分别为thumm01~thumm05，其中thumm01可由本地ssh连接（需连接网络tsinghua或使用VPN），登录thumm01后，使用命令`ssh thumm0x`可以跳转至其他主机。服务器资源有限，请勿用于课程无关任务。
    -->
- 本地SSH客户端的选择
  - 本课程使用Secure Shell(SSH)协议连接远程服务器。MacOS, Linux用户可以使用Terminal, Windows用户可以下载Miniconda、MobaXterm、XShell等软件。考虑到后续实验，我们推荐在VSCode、PyCharm等IDE中部署远程连接，方便代码编写与调试。
- 报告提交要求

- 将命令、关键代码（文本）、结果截图放入报告，实验报告需为pdf 格式，连同代码文件一同打包成压缩文件（命名为`学号_姓名_实验一.*`，例如：`2021200000_张三_实验一.zip`），最后提交到网络学堂。压缩文件中文件目录应为：

```
.
└── 学号_姓名_实验一.pdf # 实验报告
└── code # 代码文件夹
    └── code_file1
    └── code_file2
    └── ...
```

- 迟交作业一周以内，以50% 比例计分；一周以上不再计分。一经发现抄袭情况（包括往届），零分处理。

三、Linux常用命令

任务1. 使用ssh远程登录服务器 （0.5 分）

课程服务器的地址是**10.103.9.11**，使用`ssh student_id@10.103.9.11`命令即可登录服务器，其中student_id替换为学号，密码也是学号。

```
$ ssh 2019211199@10.103.9.11
2019211199@10.103.9.11's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.12.9-041209-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

2019211199@thumm01:~$
```

任务2. 配置免密登录 （0.5 分）

服务器每次登录都需要输入密码, 对此我们可以配置免密登录，原理是将本地主机的公钥保存在服务器，每次登录时主机和服务器通过公钥验证身份，因此不再需要输入密码。

生成公钥和私钥

使用ssh-keygen在个人机器上生成公钥和私钥，存放的位置一般不需要改。

```
szxie at ubuntu:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/dsjxtjc/2019211199/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/dsjxtjc/2019211199/.ssh/id_rsa.
Your public key has been saved in /home/dsjxtjc/2019211199/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:pDlFzmQA+bFtVlcSwH3hqMT9Du/qjs7rMu7eXb9yZls 2019211199@thumm01
The key's randomart image is:
+---[RSA 2048]----+
|    .o..+..oooo. |
|    . .* .o.o+.  |
|     . +=.o.o..  |
|      o=+. . .   |
```

```
|      +oS . . .  |
|       .     +   |
|              + E|
|        oo o + =.|
|       ++=B+=.*o+|
+----[SHA256]-----+
```

将公钥内容复制到服务器

使用ssh-copy-id命令将本地的公钥（localhost:~/.ssh/id_rsa.pub）添加到远程服务器的认证列表（server:~/.ssh/authorized_keys）。

```
szxie at ubuntu$ ssh-copy-id -i ~/.ssh/id_rsa.pub 2019211199@thumm01
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
"/home/dsjxtjc/2019211199/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to
filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are
prompted now it is to install the new keys
2019211199@thumm01's password:

Number of key(s) added:        1

Now try logging into the machine, with:   "ssh '2019211199@thumm01'"
and check to make sure that only the key(s) you wanted were added.
```

Windows用户可能找不到`ssh-copy-id`命令，此时需要手动拷贝本地公钥文件到服务器中

- 在执行`ssh-keygen`命令后得到公钥存放地址，比如给出的示例为`Enter file in which to save the key (/home/dsjxtjc/2019211199/.ssh/id_rsa):`
- 登录服务器后，执行命令`mkdir ~/.ssh`，新建.ssh文件夹
- 使用scp命令将本地公钥发送至服务器，`scp <key_path> student_id@10.103.9.11:~/.ssh/authorized_keys`

如果server:~/.ssh/authorized_keys文件已经存在，也可以手动把本地公钥内容粘贴到服务器该文件内

任务3. pwd、mkdir、cd命令

查看当前目录

```
2019211199@thumm01:~$ pwd
/home/dsjxtjc/2019211199
```

创建新目录

```
2019211199@thumm01:~$ mkdir dir_name
2019211199@thumm01:~$ ls
dir_name
```

进入新目录

```
2019211199@thumm01:~$ cd dir_name
2019211199@thumm01:~/dir_name$ pwd
/home/dsjxtjc/2019211199/dir_name
```

退出回到上级目录

```
2019211199@thumm01:~/dir_name$ cd ..
2019211199@thumm01:~$ pwd
/home/dsjxtjc/2019211199
```

任务4. cp、vim、nano、ls、mv、rm命令

使用vim创建一个文件file.txt，**在命令模式下输入i 切换到插入模式**，输入内容'hello world'，**按ESC返回命令模式**，输入:wq保存并退出。

```
2019211199@thumm01:~$ vim file.txt
2019211199@thumm01:~$ ls
dir_name  file.txt
```

查看文件内容

```
2019211199@thumm01:~$ cat file.txt
helloworld
```

拷贝文件file.txt, 生成新的文件new_file.txt

```
2019211199@thumm01:~$ cp file.txt new_file.txt
2019211199@thumm01:~$ ls
dir_name  file.txt  new_file.txt
```

给新文件重命名

```
2019211199@thumm01:~$ mv new_file.txt new_file_renamed.txt
2019211199@thumm01:~$ ls
dir_name  file.txt  new_file_renamed.txt
```

删除file.txt

```
2019211199@thumm01:~$ rm file.txt
2019211199@thumm01:~$ ls
dir_name  new_file_renamed.txt
```

查看文件详细信息

```
2019211199@thumm01:~$ ls -l
total 8
drwxr-xr-x 2 2019211199 dsjxtjc 4096 Sep 20 16:02 dir_name
-rw-r--r-- 1 2019211199 dsjxtjc   11 Sep 20 16:12 new_file_renamed.txt
```

任务5. cat、head、scp、awk、grep等文本处理命令 （0.5 分）

拷贝数据集wc_dataset.txt（约13MB）到用户目录下

```
2019211199@thumm01:~$ cp /home/dsjxtjc/wc_dataset.txt ./
2019211199@thumm01:~$ ls
dir_name  new_file_renamed.txt  wc_dataset.txt
```

wc_dataset是一个包含2683500个单词的大数据集，每个单词占据一行。使用指令对该数据集进行操作。

head、tail命令

head/tail用于查看文件头部/尾部的内容，默认最多显示十行

```
2019211199@thumm01:~$ head wc_dataset.txt
chapter
i
down
the
rabbit
hole
alice
was
beginning
to
2019211199@thumm01:~$
```

也可以通过添加参数-n来设定显示的行数

```
2019211199@thumm01:~$ head -n 5 wc_dataset.txt
chapter
i
down
the
rabbit
2019211199@thumm01:~$
```

head和tail可以结合，可以查看文件中任意几行的内容。例如我们要查看wc_dataset.txt中6-10行，可以这样做

```
2019211199@thumm01:~$ head -n 10 wc_dataset.txt | tail -n 5
hole
alice
was
beginning
to
2019211199@thumm01:~$
```

重定向符'>'的使用

重定向符可以将指令执行的结果重新定向，可以将原本在控制台输出的内容输出到文件。

将wc_dataset.txt中1-5行内容保存为文件wc_1-5.txt, 将6-10行保存为wc_6-10.txt。

```
2019211199@thumm01:~$ head -n 5 wc_dataset.txt > wc_1-5.txt
2019211199@thumm01:~$ head -n 10 wc_dataset.txt | tail -n 5 > wc_6-10.txt
2019211199@thumm01:~$ ls
dir_name  new_file_renamed.txt  wc_1-5.txt  wc_6-10.txt  wc_dataset.txt
```

使用了重定向符，原先的结果输出不见了，同时可以看到多了wc_1-5.txt和wc_6-10.txt两个文件，指令的输出结果被保存在了文件中。

cat命令

查看两文件的内容

```
2019211199@thumm01:~$ cat wc_1-5.txt
chapter
i
down
the
rabbit
2019211199@thumm01:~$ cat wc_6-10.txt
hole
alice
was
beginning
to
```

`cat wc_1-5.txt wc_6-10.txt > wc_1-10.txt` 相当于合并两文件内容并保存。

```
2019211199@thumm01:~$ cat wc_1-5.txt wc_6-10.txt > wc_1-10.txt
2019211199@thumm01:~$ cat wc_1-10.txt
chapter
i
down
the
rabbit
hole
alice
was
beginning
to
```

scp命令

scp命令用来在不同主机之间传输文件，它使用的是SSH协议。

这里需要开启两个终端来查看结果，分别连接上thumm01, thumm03。需要注意的是，thumm03节点未提供外网地址，需要先登录thumm01，使用命令`ssh thumm03`跳转。

在thumm01上

```
2019211199@thumm01:~$ ls
dir_name  new_file_renamed.txt  wc_1-10.txt  wc_1-5.txt  wc_6-10.txt
wc_dataset.txt
2019211199@thumm01:~$
```

在thumm03上

```
2019211199@thumm03:~$ ls
2019211199@thumm03:~$
```

将thumm01中的wc_1-10.txt传到thumm03，其中`~/`代表用户目录

在thumm01上

```
2019211199@thumm01:~$ scp wc_1-10.txt thumm03:~/
2019211199@thumm03's password:
wc_1-10.txt                     100%   54     0.1KB/s   00:00
```

在thumm03上多出wc_1-10.txt

```
2019211199@thumm03:~$ ls
wc_1-10.txt
```

awk命令

awk是一个强大的文本分析工具，我们仅介绍一些常用功能。

基本用法

```
awk [选项参数] 'script' var=value file(s)
或
awk [选项参数] -f scriptfile var=value file(s)
```

awk指令适合处理格式规整的数据，例如`/etc/passwd`文件，它保存着Linux系统中用户的用户名以及其他信息（不包含密码），我们可以通过它了解当前主机上的用户信息，例如我将使用`awk`查看服务器用户列表中学号为2021开头的学生的个数。

要处理数据，我们首先要分析一下数据的格式

```
2019211199@thumm01:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
......
2021214323:x:1372:502::/home/dsjxtjc/2021214323:/bin/bash
2021280795:x:1373:502::/home/dsjxtjc/2021280795:/bin/bash
```

我们可以看到，数据每一行代表一个用户，开头为用户的用户名，后面为用户信息（具体代表什么不用管），每个信息使用`:`隔开。对此，我们可以依据冒号进行分割，然后取每行第一个元素（用户名），使用正则表达式匹配下看用户名是否为2021开头，如果是则输出。

要实现这个功能，我们可以使用下面的指令

```
2019211199@thumm01:~$ awk -F: '$1~"^2021"{print $1}' /etc/passwd
2021210991
2021211018
......
20212144842
```

其中-F:参数表示使用`:`作为分隔符进行分割，$1~"^2021"{print $1}中$1表示分割后第一个元素（用户名），"^2021"是一个正则表达式，表示以2021开头，`~`表示匹配，所以$1~"^2021"表示分割后第一个元素满足2021开头，那么就执行后面的指令{print $1}。

使用`wc -l`统计下有多少个这样的学号，即可知道参加本课程的21级同学的数量了。

```
2019211199@thumm01:~$ awk -F: '$1~"^2021"{print $1}' /etc/passwd | wc -l
xx
```

可以看到，一共有xx名21级同学参加本课程~~

grep命令

> grep命令用于查找文件里符合条件的字符串。如果发现某文件的内容符合所指定的范本样式，预设 grep 指令会把含有范本样式的那一列显示出来。若不指定任何文件名称，或是所给予的文件名为-，则 grep 指令会从标准输入设备读取数据。

接下来我们使用grep命令对wc_dataset.txt作分析

1. 显示以"dis"开头的单词（显示前10条）

```
2019211199@thumm01:~$ grep "^dis" wc_dataset.txt |head
disappointment
distance
disagree
distance
distance
distance
distant
dish
dishes
disgust
```

2. 反向过滤，添加参数-v。

查找wc_1-10.txt中以t字母开头的单词：

```
2019211199@thumm01:~$ grep "^t" wc_1-10.txt
the
to
```

接着添加参数-v，过滤掉以t开头的单词。

```
2019211199@thumm01:~$ grep -v "^t" wc_1-10.txt
chapter
i
down
rabbit
hole
alice
was
beginning
```

任务6. 阻塞与非阻塞时间对比 （0.5 分）

在Linux shell脚本中，指令是顺序执行的，但实际上没有相互依赖（或者说数据关联）的指令可以并行地运行而对结果没有影响。为了让同一个脚本中没有相互依赖的指令并行地执行，我们需要指定这些指令为非阻塞。

让指令不阻塞的方法是在指令最后加上'&'。

接下来边写两个脚本，第一个脚本使用阻塞的方法执行，第二个脚本使用非阻塞的方法执行，对比两个脚本的运行时间：

脚本一: shell_blocked.sh

```
#!/bin/bash
awk '$1~"^chapter"{}' wc_dataset.txt
awk '$1~"^chapter"{}' wc_dataset.txt
awk '$1~"^chapter"{}' wc_dataset.txt
awk '$1~"^chapter"{}' wc_dataset.txt
awk '$1~"^chapter"{}' wc_dataset.txt
```

脚本二：shell_unblocked.sh

```
#!/bin/bash
awk '$1~"^chapter"{}' wc_dataset.txt &
awk '$1~"^chapter"{}' wc_dataset.txt &
awk '$1~"^chapter"{}' wc_dataset.txt &
awk '$1~"^chapter"{}' wc_dataset.txt &
awk '$1~"^chapter"{}' wc_dataset.txt &
wait
```

运行这两个脚本，对比它们运行的时间

```
2019211199@thumm01:~$ vim shell_block.sh
2019211199@thumm01:~$ vim shell_unblock.sh
2019211199@thumm01:~$ time bash ./shell_block.sh

real    0m5.387s
user    0m5.323s
sys 0m0.064s

2019211199@thumm01:~$ time bash ./shell_unblock.sh

real    0m1.142s
user    0m5.521s
sys 0m0.025s
```

运行程序后，可以看到用户时间不变，都是5秒左右。但对于真实时间，脚本二是脚本一的五分之一，因此从用户的角度看脚本二运行更快。
（用户时间user time是指程序在多个核上运行时间的和，真实时间real time是现实中程序运行过去了多长时间，真实时间变短原因是每个操作不再阻塞，而是利用多个处理器核心并行计算。）

四、多主机任务处理

任务7. 多节点任务处理 （3 分）

为了充分利用集群的运算性能，我们需要将资源分配至各个节点、协调各个节点的任务、整合多个结果等等。接下来我们来控制命令在多个主机上协同运行。目前集群节点有thumm01，thumm03~thumm05。

集群主机之间免密登录配置 （1.5 分）

请你用linux shell写一个脚本auto_autho.sh，实现各节点之间的免密登录。即实现thumm01分别到thumm03~thumm05的免密登录，使得运行该脚本后，可以通过ssh thumm0**X**从thumm01免密登录到**X**号节点。

思路：这个脚本做的事情是在thumm01上生成2个节点的公钥和私钥，然后把所有公钥加入到authorized_keys中，然后把各自的公钥私钥以及authorized_keys分发到各个节点。

多结点任务处理 （1.5 分）

请仿照wc_dataset.txt，制作2G左右的数据集（比如将wc_dataset.txt重复拼接）。在多主机运行一个简单的词频统计任务并汇总（即每个单词出现多少次），对比单机处理和多机处理的差异，可以包括任务执行结果、延迟等方面。

Bonus（1分）

尝试提出一种可以加快多节点处理速度的方法并验证。需要注意的是，由于各节点间带宽存在实时波动，请在验证中论证“提出的方法不是因为网络波动带来的虚假数值增益”。