西南科技大学计算机科学与技术学院

实验报告

实 验 名 称 Linux 环境安装配置

实 验 地 点 东 11C225 大学计算机实验室

实 验 日 期 2024/10/10

指 导 教 师 韦勇

学 生 班 级

学 生 姓 名

学生学号

提 交 日 期 2024/10/12

网络环境配置

一、实验目的

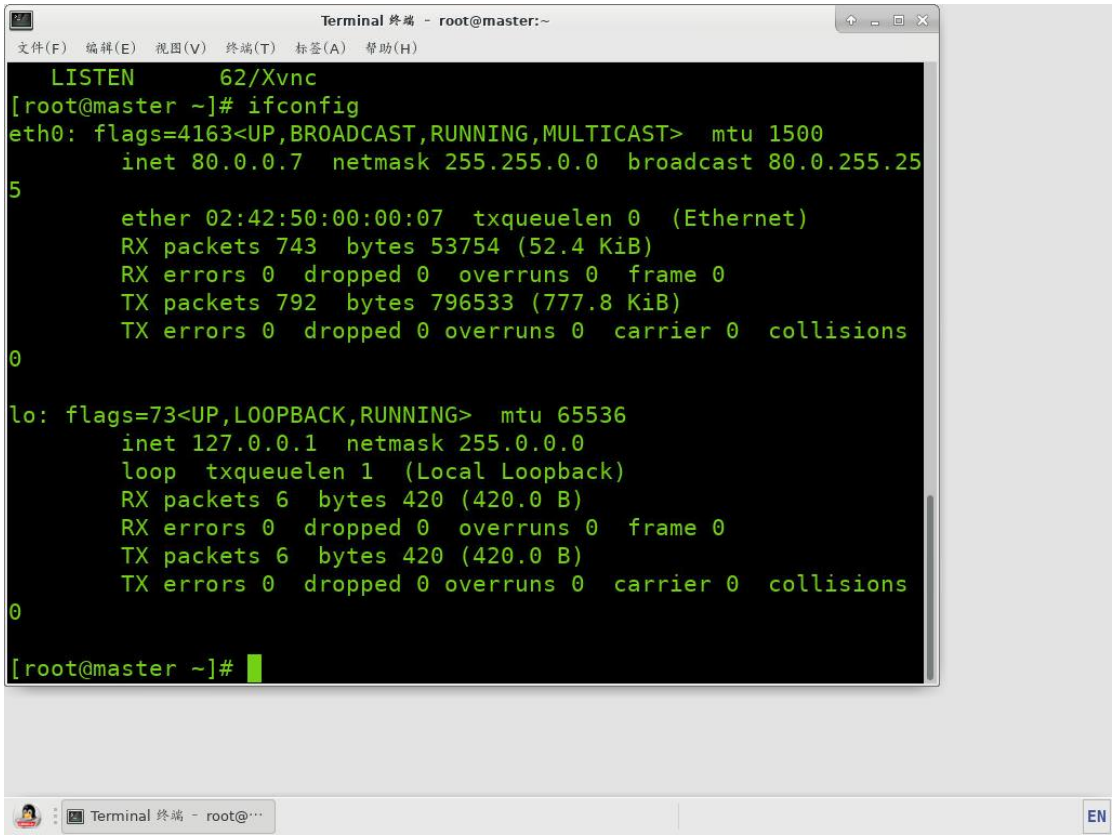
了解 Linux 下的网络功能，理解 Linux 强大的网络功能。

二、实验设计

请写出命令查看网络 IP 地址和路由信息，并解释命令的输出结果。

三、实验记录

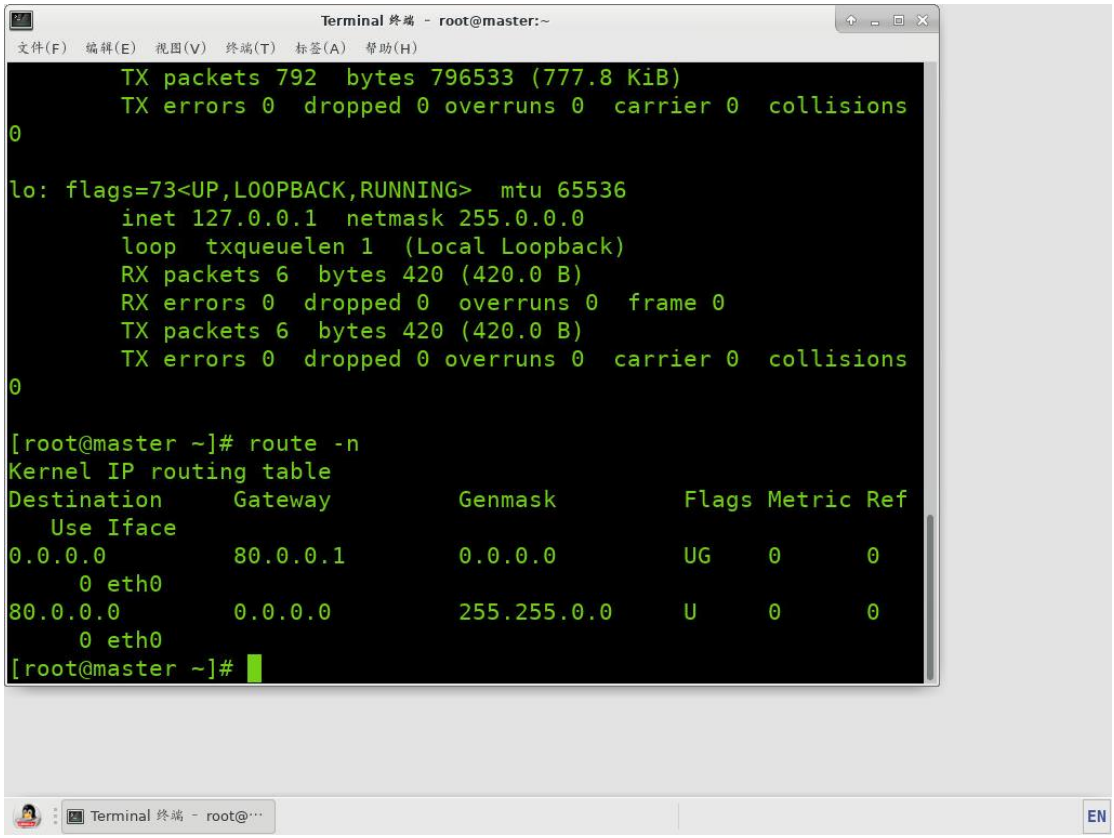
Linux 下查看网卡设置为 ifconfig



在这个结果中，eth0 为第一块物理网卡，lo 是一个本地回环设备，它是一个软件网络设备，与这个设备的通信不会传输到物理网络上，从输出结果可以看到，eth0 已经配好了 IPv4 地址为 80.0.0.7，子网掩码为 255.255.0.0，广播地址（用于发送给同一网络内所有设备的广播消息）为 80.0.255.255。

RX packets:743 表示接收到的数据包数量 TX packets:15783 表示发送的数据包数量。

Route 命令用于查看、设置网络路由，-n 参数表示直接显示为 ip 地址。route -n 显示了 Linux 系统的内核 IP 路由表，路由表决定了数据包从一台主机发送到另一台主机时所走的路径。



Destination：目的网络地址或主机地址。0.0.0.0，意味着任何目的地。

Gateway：到达目的网络的下一跳地址。0.0.0.0，表示不通过网关。

Genmask：目的网络的子网掩码，用于确定目的网络的地址范围。对于默认路由来说，子网掩码总是 0.0.0.0。

Flags：路由标志，其中 U 表示路由是上行的（active），G 表示使用网关。

Metric：路由的度量值，用于多条路由规则中选择成本最低的路由。

Ref：路由的引用计数，表示此路由被其他路由或者系统配置引用的次数。

Use：此路由被使用的计数器，表示路由被查找和使用的次数。

Iface：数据包将通过的网络接口。

0.0.0.0 80.0.0.1 0.0.0.0 UG 0 0 0 eth0 是默认路由，表示所有非本地网络的流量都将通过这个路由，流量将通过 eth0 接口被发送到 80.0.0.1。

80.0.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth0 是一条直接路由，表示目的地址为 80.0.0.0 的网络将直接发送到 eth0 接口，不经过任何网关。

这个结果表示所有目的地不是本网络（80.0.0.0 网络）的流量都将通过网关 80.0.0.1 离开，而目的地是 80.0.0.0 网络的流量将直接在本地网络上发送。

四、实验思考或体会

实验思考：如何确认实验环境中提供 TCP 22 端口的服务进程？如何确定该进程的应用程序属于哪个安装包？写出步骤和结果。

Netstat 命令用于显示各种网络相关信息，如接口状态。使用—help 查看手册。

```
Terminal 终端 - root@master:~
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
[root@master ~]# netstat --help
usage: netstat [-vWeenNcCF] [<Af>] -r         netstat {-V|--versi
on|-h|--help}
       netstat [-vWnNcaeol] [<Socket> ...]
       netstat { [-vWeenNac] -I[<Iface>] | [-veenNac] -i | [-cnNe
] -M | -s [-6tuw] } [delay]

        -r, --route              display routing table
        -I, --interfaces=<Iface> display interface table for <Ifa
ce>
        -i, --interfaces         display interface table
        -g, --groups             display multicast group membersh
ips
        -s, --statistics         display networking statistics (l
ike SNMP)
        -M, --masquerade         display masqueraded connections

        -v, --verbose            be verbose
        -W, --wide               don't truncate IP addresses
        -n, --numeric            don't resolve names
        --numeric-hosts          don't resolve host names
        --numeric-ports          don't resolve port names
```

-l 列出监听状态的端口，-n 显示出 ip，-t 查看所有 tcp 端口，-p 显示进程的 PID 和名称，-u 表示显示 UDP 端口。

使用 netstat -tunlp |grep 22，它会列出所有监听端口 22（通常是 SSH 服务使用的端口）的 TCP 和 UDP 套接字，以及相关的进程信息。

```
Terminal 终端 - root@master:~
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
nnected)
        -F, --fib                display Forwarding Information B
ase (default)
        -C, --cache              display routing cache instead of
 FIB
        -Z, --context            display SELinux security context
 for sockets

  <Socket>={-t|--tcp} {-u|--udp} {-U|--udplite} {-S|--sctp} {-w|-
-raw}
           {-x|--unix} --ax25 --ipx --netrom
  <AF>=Use '-6|-4' or '-A <af>' or '--<af>'; default: inet
  List of possible address families (which support routing):
    inet (DARPA Internet) inet6 (IPv6) ax25 (AMPR AX.25)
    netrom (AMPR NET/ROM) ipx (Novell IPX) ddp (Appletalk DDP)
    x25 (CCITT X.25)
[root@master ~]# netstat -tunlp | grep 22
tcp        0      0 0.0.0.0:22              0.0.0.0:*
   LISTEN      574/sshd
tcp6       0      0 :::22                   :::*
   LISTEN      574/sshd
[root@master ~]#
```

发现 22 端口的进程 PID 是 574，进程名称为 sshd。

使用 ls -l /proc/574/exe 命令查看进程的可执行文件路径。在 Linux 系统中，/proc 是一个虚拟的文件系统，它包含了运行中的内核和进程的信息。每个正在运行的进程在 /proc 文件系统中都有一个以其 PID 命名的目录。在该目录下，包含了许多与该进程相关的文件和子目录，其中之一就是 exe 符号链接。

```
[root@master ~]# ls -l /proc/574/exe
lrwxrwxrwx 1 root root 0 10月  10 18:56 /proc/574/exe -> /usr/sbin
/sshd
[root@master ~]#
```

得到进程 574 的可执行文件位于/usr/sbin/sshd

使用 rpm -qf /usr/sbin/sshd 查询/usr/sbin/sshd 属于哪个安装包。

```
[root@master ~]# rpm -qf /usr/sbin/sshd
openssh-server-7.4p1-21.el7.x86_64
```

得到安装包为 openssh-server-7.4p1-wq.el7.x86_64

体会：

通过本次实验，我深入了解了 Linux 系统在网络配置和管理方面的强大功能。使用 ip 和 route 命令，我能够轻松查看和理解网络接口的配置以及系统的路由策略。这些命令不仅提供了丰富的网络信息，而且它们的输出结果非常详细，有助于诊断网络问题和优化网络性能。Linux 的网络功能确实强大，它为网络管理和故障排查提供了有力的工具。通过这些基本的命令，即使是网络新手也能够快速掌握网络配置的基础知识，并逐步深入了解更高级的网络概念和应用。

网络服务管理

一、实验目的

验证两种网络服务的管理办法，理解 linux 强大的网络功能。

二、实验设计

确定 mysql 服务的状态，并重启该服务。使用了`systemctl status mysql`命令来检查 MySQL 服务的状态，通过`systemctl restart mysql`命令成功重启了 MySQL 服务。

三、实验记录

使用 systemctl status mysql 命令确定 mysql 服务状态，systemctl 是 systemd 系统和服务管理器的命令行工具，而 systemd 是现代 Linux 系统中用于初始化系统组件和管理系统进程的标准。

可以看到 mysql 处于 active 状态。

```
[root@master ~]# systemctl status mysql
● mysql.service - LSB: start and stop MySQL
   Loaded: loaded (/etc/rc.d/init.d/mysql; bad; vendor preset: di
sabled)
   Active: active (running) since 四 2024-10-10 19:26:14 CST; 6mi
n ago
     Docs: man:systemd-sysv-generator(8)
   CGroup: /docker/9c158bf77695bdf0229dc78ab0dbd247995a5a29a236e2
2256dfbb76fca25426/system.slice/mysql.service
           ├─645 /bin/sh /usr/bin/mysqld_safe --datadir=/var/l...
           └─762 /usr/sbin/mysqld --basedir=/usr --datadir=/va...

10月 10 19:26:13 master systemd[1]: Starting LSB: start and s...
10月 10 19:26:14 master mysql[633]: Starting MySQL[  OK  ]
10月 10 19:26:14 master systemd[1]: Started LSB: start and st...
Hint: Some lines were ellipsized, use -l to show in full.
[root@master ~]#
```

也可以使用 ps aux |grep mysql 命令查看 mysql 进程状态，ps 显示当前运行在 Linux 系统中的进程信息，a 显示所有终端的进程，包括那些没有控制终端的进程。u 以用户易读的格式显示进程信息，包括进程的用户、CPU 使用率、内存使用率等。x 显示没有控制终端的进程。即使没有终端，这些进程也会被列出。

结果显示了 MySQL 服务的主进程和其子进程状态。

```
Terminal 终端 - root@master:~
文件(F)  编辑(E)  视图(V)  终端(T)  标签(A)  帮助(H)
[root@master ~]# vim /etc/xinetd.d/telnet
[root@master ~]# systemctl start xinetd
[root@master ~]# netstat -tunlp | grep 23
tcp6       0      0 :::23                   :::*
   LISTEN      1211/xinetd
[root@master ~]# systemctl stop xinetd
[root@master ~]# netstat -tunlp | grep 23
[root@master ~]#
[root@master ~]#
[root@master ~]#
[root@master ~]#
[root@master ~]# ps aux | grep mysql
root        645  0.0  0.0  11820  1612 ?        S    19:26   0:00
 /bin/sh /usr/bin/mysqld_safe --datadir=/var/lib/mysql --pid-file
=/var/lib/mysql/master.pid
mysql       762  0.0  0.0 448952 40488 ?        Sl   19:26   0:00
 /usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql --plugi
n-dir=/usr/lib64/mysql/plugin --user=mysql --log-error=master.err
 --pid-file=/var/lib/mysql/master.pid
root       1271  0.0  0.0 112712   980 pts/0    S+   19:32   0:00
 grep --color=auto mysql
[root@master ~]#
Terminal 终端 - root@···                                        EN
```

使用 systemctl restart mysql 重启 mysql 进程，重启后 systemctl status mysql 确认状态

```
Terminal 终端 - root@master:~
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
sabled)
   Active: active (running) since 四 2024-10-10 19:26:14 CST; 6mi
n ago
     Docs: man:systemd-sysv-generator(8)
   CGroup: /docker/9c158bf77695bdf0229dc78ab0dbd247995a5a29a236e2
2256dfbb76fca25426/system.slice/mysql.service
           ├─645 /bin/sh /usr/bin/mysqld_safe --datadir=/var/l...
           └─762 /usr/sbin/mysqld --basedir=/usr --datadir=/va...

10月 10 19:26:13 master systemd[1]: Starting LSB: start and s...
10月 10 19:26:14 master mysql[633]: Starting MySQL[  OK  ]
10月 10 19:26:14 master systemd[1]: Started LSB: start and st...
Hint: Some lines were ellipsized, use -l to show in full.
[root@master ~]# systemctl is-active mysql
active
[root@master ~]# systemctl is-enabled mysql
mysql.service is not a native service, redirecting to /sbin/chkco
nfig.
Executing /sbin/chkconfig mysql --level=5
enabled
[root@master ~]# systemctl restart mysql
[root@master ~]#
Terminal 终端 - root@…
EN
```

也可以使用 service mysql restart 重启服务。

```
[root@master ~]# service mysql restart
Shutting down MySQL.                                       [  OK
 ]
Starting MySQL..                                           [  OK
 ]
[root@master ~]#
```

```
Terminal 终端 - root@master:~
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
[root@master ~]# service mysql restart
Shutting down MySQL.                                       [  OK
 ]
Starting MySQL..                                           [  OK
 ]
[root@master ~]# systemctl status mysql
● mysql.service - LSB: start and stop MySQL
   Loaded: loaded (/etc/rc.d/init.d/mysql; bad; vendor preset: di
sabled)
   Active: active (exited) since 四 2024-10-10 19:33:58 CST; 41s
ago
     Docs: man:systemd-sysv-generator(8)
  Process: 1306 ExecStop=/etc/rc.d/init.d/mysql stop (code=exited
, status=0/SUCCESS)
  Process: 1325 ExecStart=/etc/rc.d/init.d/mysql start (code=exit
ed, status=0/SUCCESS)

10月 10 19:33:56 master systemd[1]: Starting LSB: start and s...
10月 10 19:33:58 master mysql[1325]: Starting MySQL..[  OK  ]
10月 10 19:33:58 master systemd[1]: Started LSB: start and st...
Hint: Some lines were ellipsized, use -l to show in full.
[root@master ~]#
```

四、实验思考或体会

实验思考：一个网络服务可以同时被 systemd 和 xinetd 管理吗？举例说明。

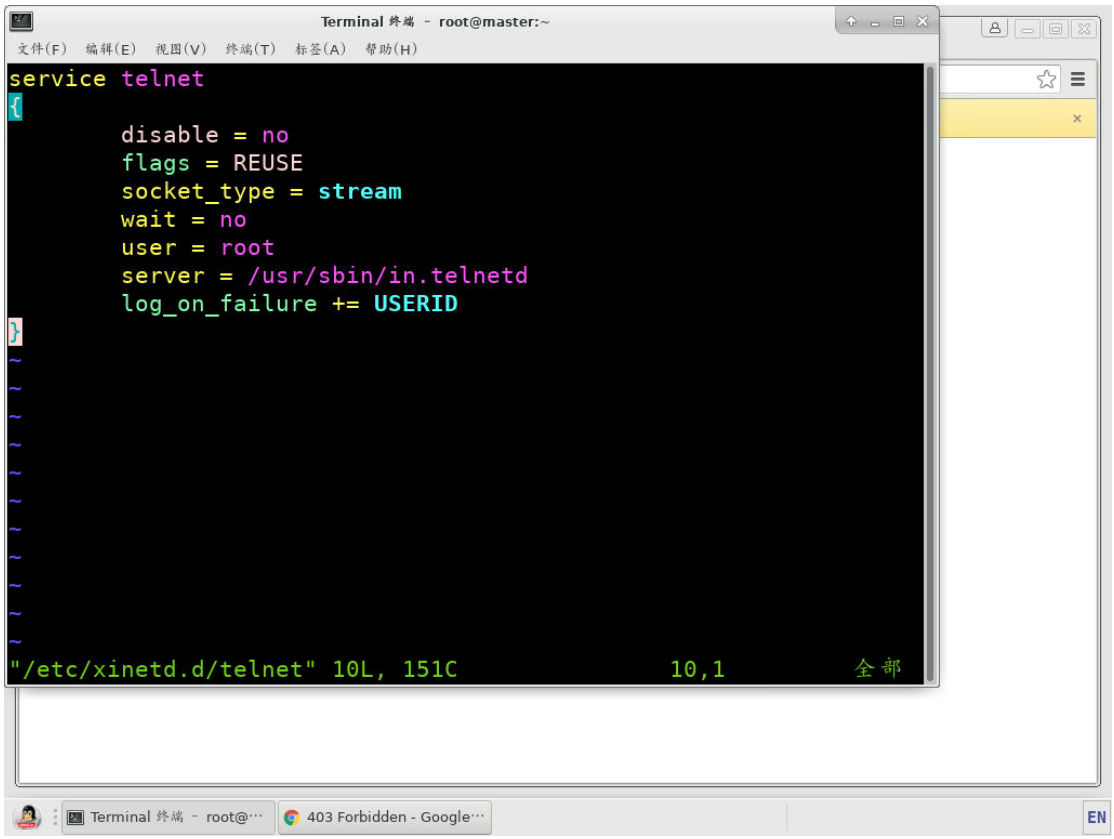
不能。以 telnet 为例。

安装 xinetd、telnet、telnet-server 软件包

```
Terminal 终端 - root@master:~
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
[root@master ~]# rpm -ivh /root/experiment/file/xinetd-2.3.15-13.
el7.x86_64.rpm
Preparing...
################################# [100%]
Updating / installing...
   1:xinetd-2:2.3.15-13.el7
################################# [100%]
[root@master ~]# rpm -ivh /root/experiment/file/telnet-0.17-64.el
7.x86_64.rpm
Preparing...
################################# [100%]
Updating / installing...
   1:telnet-1:0.17-64.el7
################################# [100%]
[root@master ~]# rpm -ivh /root/experiment/file/telnet-server-0.1
7-64.el7.x86_64.rpm
Preparing...
################################# [100%]
Updating / installing...
   1:telnet-server-1:0.17-64.el7
################################# [100%]
[root@master ~]#
```

配置 telnet-server，编辑/etc/xinetd.d/telnet 文件，定义了如何启动和管理 telnet

服务。



disable = no 表明该服务没有被禁用。flags = REUSE 表示允许服务快速重启。socket_type = stream 指定了 socket 的类型为流式(socket_type)，这是 TCP/IP 协议中的一种可靠的数据传输服务。wait = no 表明服务应该并行运行，而不是串行。user = root 定义了运行服务的用户是 root。server = /usr/sbin/in.telnetd 指定了服务的服务器程序及其路径。当有新的 telnet 连接请求时，xinetd 会启动这个程序来处理连接。log_on_failure += USERID 定义了在服务启动失败时记录日志的级别。USERID 表示日志中将包含失败尝试的用户 ID 信息。

启动 telnet 服务（systemctl start xinetd）并监听 23 端口，看 telnet 服务是否随着 xinetd 服务启动而启动，发现是的。

```
[root@master ~]# vim /etc/xinetd.d/telnet
[root@master ~]# vim /etc/xinetd.d/telnet
[root@master ~]# systemctl start xinetd
[root@master ~]# netstat -tunlp | grep 23
tcp6       0      0 :::23                   :::*
   LISTEN      1466/xinetd
```

使用 systemctl 关闭 httpd，发现 23 端口仍在工作，从而验证不能。

```
[root@master ~]# systemctl stop httpd
[root@master ~]# systemctl status telnet
Unit telnet.service could not be found.
[root@master ~]# netstat -tunlp |grep 23
tcp6       0      0 :::23                   :::*
   LISTEN      1776/xinetd
[root@master ~]#
```

体会：

在本次实验中，我通过实际操作深入体验了 Linux 系统下两种主要的网络服务管理方法：systemd 和 xinetd。

通过进一步的探索和学习，我了解到一个网络服务同时被 systemd 和 xinetd 管理虽然理论上可能，但实际上这样做会导致冲突和混乱。systemd 和 xinetd 都旨在管理系统服务，但它们的管理机制不同。systemd 是现代 Linux 系统中的标准服务管理器，而 xinetd 是一个传统的超级守护进程，用于管理一些轻量级的网络服务。在现代系统中，大多数服务都通过 systemd 来管理，而 xinetd 的使用越来越少。

这个实验不仅加深了我对 Linux 网络服务管理的理解，也让我认识到了技术发展的潮流。随着 systemd 逐渐成为主流，它提供了更加强大和统一的服务管理方式，使得服务的启动、停止、重启和监控变得更加简单和高效。

网络访问控制

一、实验目的

验证网络访问控制的实现方法，理解 linux 强大的网络功能。

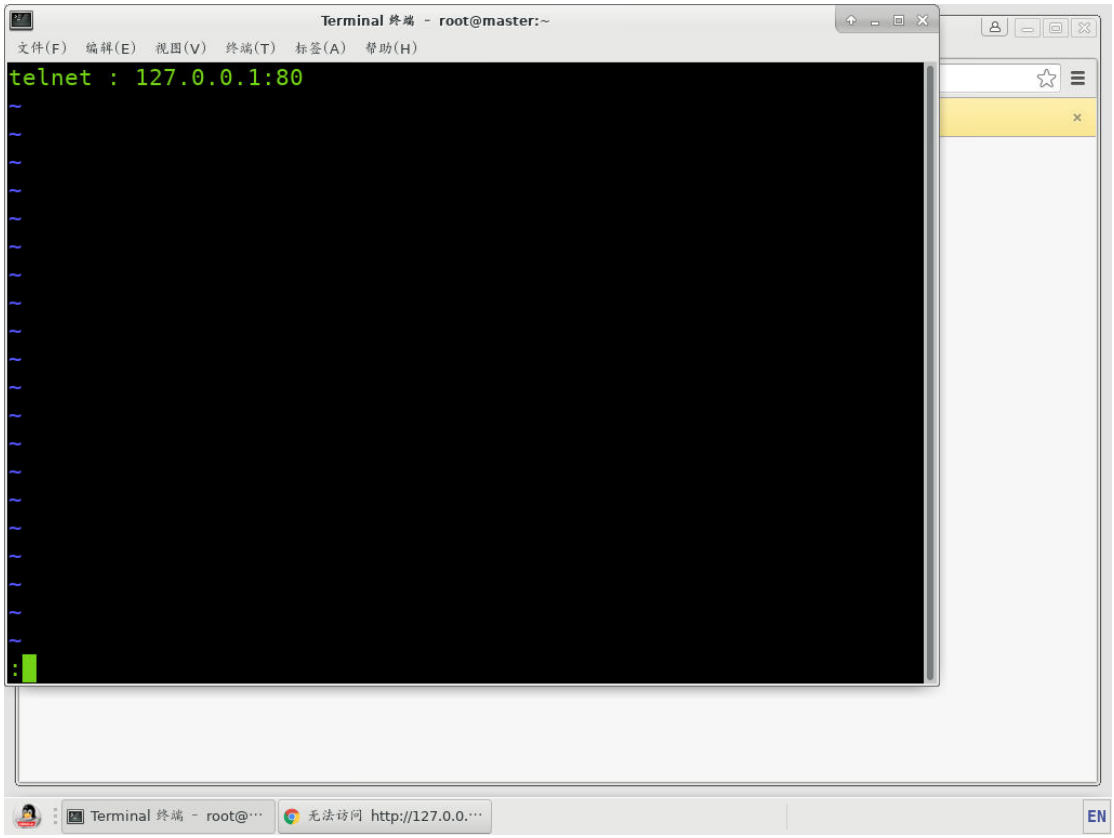
二、实验设计

试配置仅拒绝从某 IP 的实验环境（“节点 1” 或 “节点 2”）中访问本机 telnet 服务（“主节点”）。(提示需要该同学实验环境中的 IP 地址加如/etc/hosts.deny 中，注意该文件的格式。)

三、实验记录

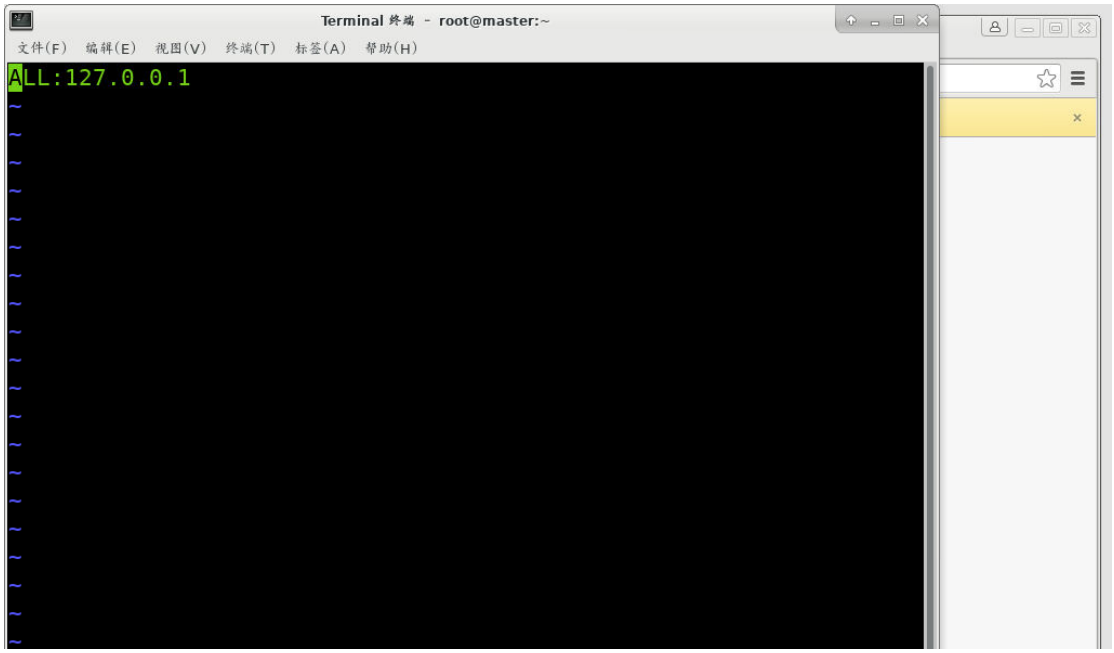
仅拒绝特定 IP 访问主机上的 telnet 服务，可以使用 hosts.deny 文件来实现，hosts.allow 和 hosts.deny 文件用于控制对本地服务的网络访问。这两个文件共同工作，以决定哪些主机可以访问特定的服务。hosts.deny 文件位于/etc 目录下

使用 vim /etc/hosts.deny，为 telnet 服务添加一行，指定想要拒绝访问的 IP 地址（127.0.0. 1）



保存退出后，hosts.allow 文件用于指定允许访问服务的主机，如果想要允许其他所有主机访问 telnet 服务，除了在 hosts.deny 中指定的 IP，则需要在 hosts.allow 中添加相应行。

Vim /etc/hosts.allow



修改完后需要重启 telnet 服务，systemctl restart xinetd

```
[root@master ~]# vim /etc/hosts.deny
[root@master ~]# vim /etc/hosts.allow
[root@master ~]# vim /etc/hosts.deny
[root@master ~]# systemctl restart xinetd
[root@master ~]#
```

验证：

```
[root@master ~]# telnet localhost
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Connection closed by foreign host.
[root@master ~]#
```

四、实验思考或体会

实验思考：当 hosts.all 和 hosts.deny 中的配置有矛盾（例如对同一网络服务，一个配置允许访问，一个配置拒绝访问）的时候，网络访问控制如何进行？

hosts.allow 优先。如果一个服务在 hosts.allow 件中被明确允许，即使它也在 hosts.deny 文件中被拒绝，该服务仍然可以被访问。

以 telnet 服务为例，/etc/hosts.allow 文件中有 telnet: 127.0.0.1，意味着本机被允许访问 telnet 服务。/etc/hosts.deny 文件中有：telnet: 127.0.0.1，意味着本机被拒绝访问 telnet 服务。在这种情况下，由于本机在 hosts.allow 中被明确允许，它将能够访问 telnet 服务，即使它也在 hosts.deny 中被拒绝。：

体会：

在本次实验中，我通过实际操作深入体验了 Linux 系统下网络访问控制的实现方法，特别是通过`/etc/hosts.deny`文件来限制特定 IP 地址对本地服务的访问。通过这个实验，我更加直观地感受到了 Linux 在网络安全管理方面的强大和灵活性。

在实验过程中，我首先了解了`hosts.allow`和`hosts.deny`文件的作用和配置方法。我学习了如何在`hosts.deny`文件中添加规则来拒绝特定 IP 地址访问本机的 telnet 服务。这个过程中，我注意到了文件格式的重要性，例如需要正确地指定服务名称和 IP 地址，以及确保文件中没有语法错误。

此外，我还学习了当`hosts.allow`和`hosts.deny`文件中的配置出现矛盾时，系统如何处理这些规则。我了解到，如果一个 IP 地址在`hosts.allow`中被明确允许，即使它在`hosts.deny`中被拒绝，该 IP 地址仍然可以访问服务。这个规则让我意识到了在配置网络安全规则时，需要全面考虑所有可能的配置组合，以避免出现意外的访问控制结果。

Linuxer 的服务端配置

一、实验目的

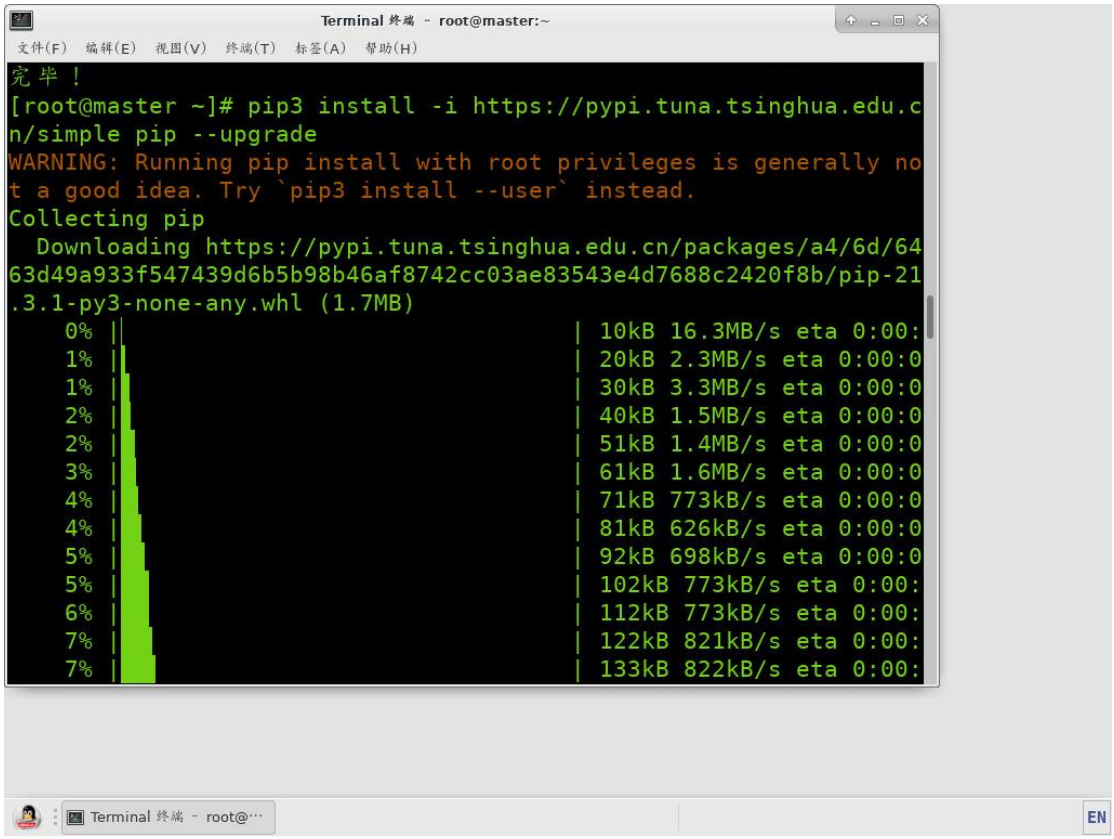
通过案例系统服务器端的配置掌握 linux 环境服务配置的基本方法。

二、实验设计

先安装好 linuxer，然后对教师账号、学生账号设置登陆权限，进入学生账号进行实验报告上传，教师账号批阅后，返回学生账号查看成绩。

三、实验记录

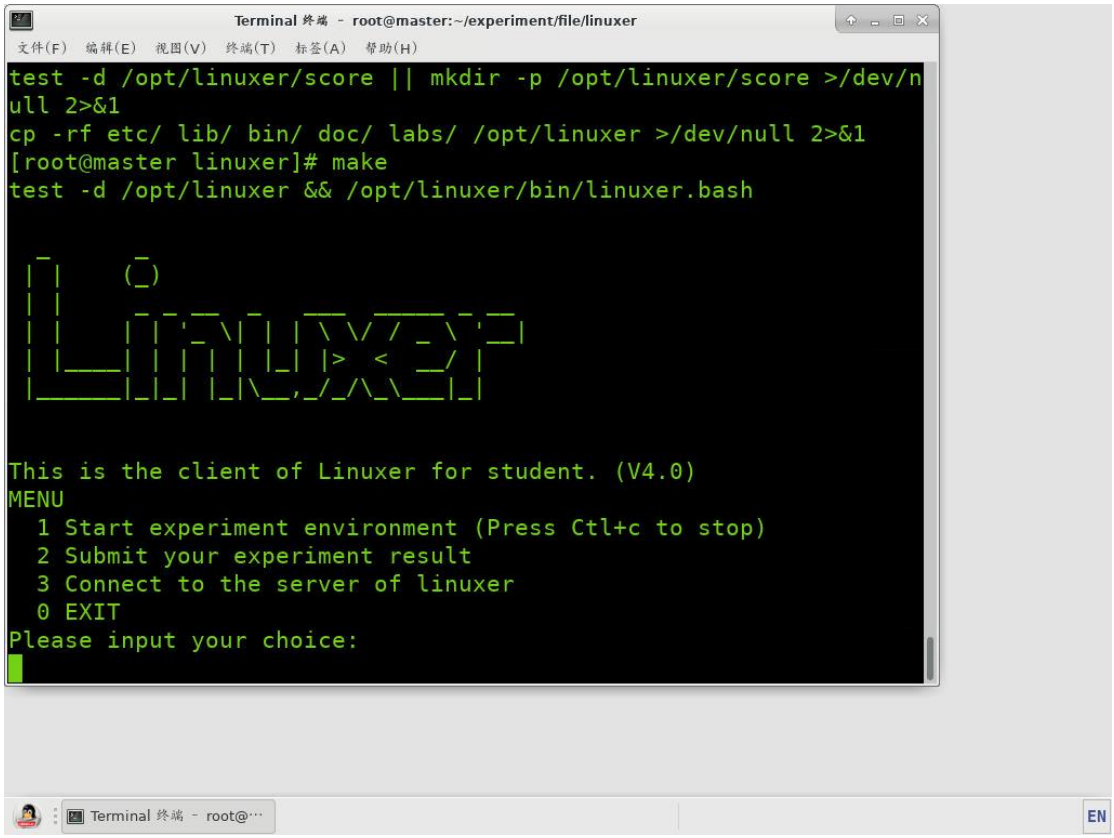
实验二任务三：配置安装源后，安装依赖项 python、juptyer 以及 bash_kernel、wput 并更新



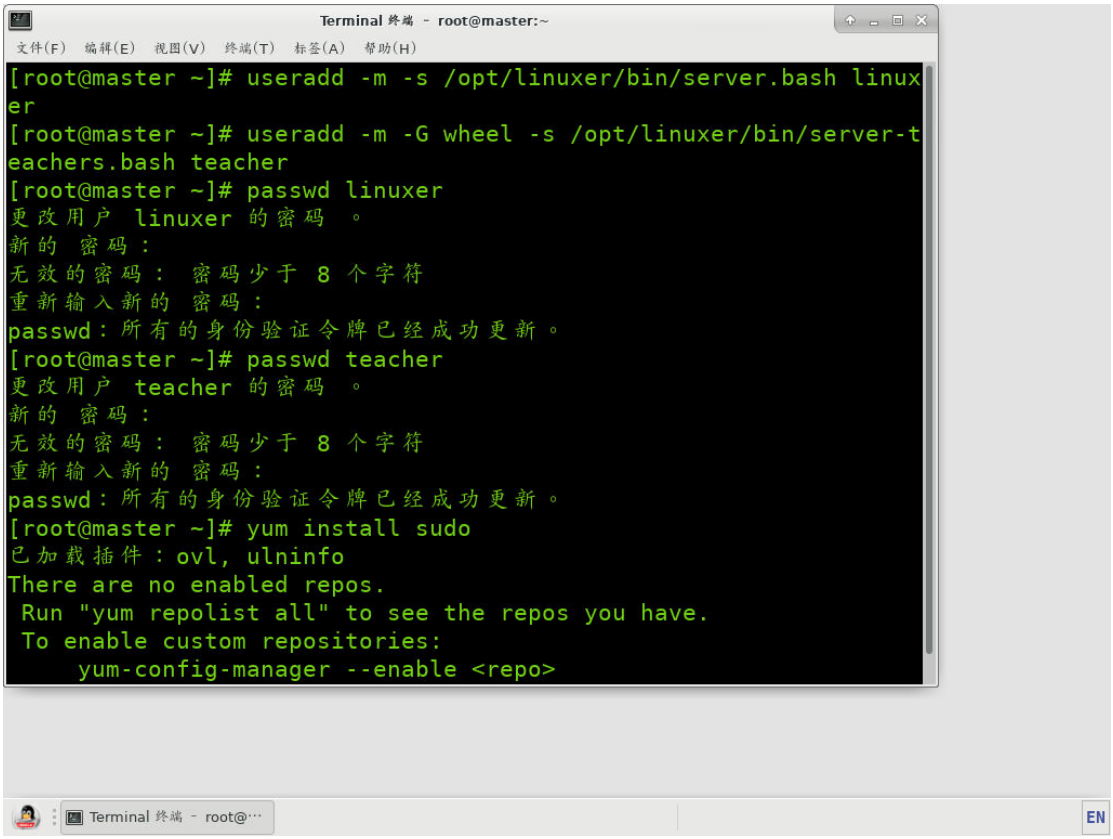
```
Terminal 终端 - root@master:~
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
Collecting zipp>=0.5
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/bd/df/d4
a4974a3e3957fd1c1fa3082366d7fff6e428ddb55f074bf64876f8e8ad/zipp-3
.6.0-py3-none-any.whl (5.3 kB)
Using legacy 'setup.py install' for contextvars, since package 'w
heel' is not installed.
Installing collected packages: zipp, typing-extensions, six, ipyt
hon-genutils, decorator, traitlets, pyrsistent, importlib-metadat
a, attrs, wcwidth, tornado, pyzmq, python-dateutil, pyparsing, py
cparser, ptyprocess, parso, nest-asyncio, jupyter-core, jsonschem
a, immutables, entrypoints, webencodings, pygments, prompt-toolki
t, pickleshare, pexpect, packaging, nbformat, MarkupSafe, jupyter
-client, jedi, contextvars, cffi, backcall, async-generator, test
path, sniffio, pandocfilters, nbclient, mistune, jupyterlab-pygme
nts, jinja2, ipython, idna, defusedxml, dataclasses, bleach, argo
n2-cffi-bindings, websocket-client, urllib3, terminado, Send2Tras
h, pytz, prometheus-client, nbconvert, ipykernel, charset-normali
zer, certifi, argon2-cffi, anyio, requests, notebook, jupyter-ser
ver, json5, babel, widgetsnbextension, nbclassic, jupyterlab-widg
ets, jupyterlab-server, comm, jupyterlab, jupyter-console, ipywid
gets, jupyter
```

```
Terminal 终端 - root@master:~
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
ernel) (1.6.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local
/lib/python3.6/site-packages (from jupyter-client->ipykernel->bas
h_kernel) (2.9.0.post0)
Requirement already satisfied: pyzmq>=13 in /usr/local/lib64/pyth
on3.6/site-packages (from jupyter-client->ipykernel->bash_kernel)
 (25.1.2)
Requirement already satisfied: parso<0.8.0,>=0.7.0 in /usr/local/
lib/python3.6/site-packages (from jedi<=0.17.2,>=0.10->ipython>=5
.0.0->ipykernel->bash_kernel) (0.7.1)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.
6/site-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.
0->ipython>=5.0.0->ipykernel->bash_kernel) (0.2.13)
Installing collected packages: bash-kernel
Successfully installed bash-kernel-0.9.3
WARNING: Running pip as the 'root' user can result in broken perm
issions and conflicting behaviour with the system package manager
. It is recommended to use a virtual environment instead: https:/
/pip.pypa.io/warnings/venv
[root@master ~]# python3 -m bash_kernel.install
Installing IPython kernel spec
[root@master ~]#
```

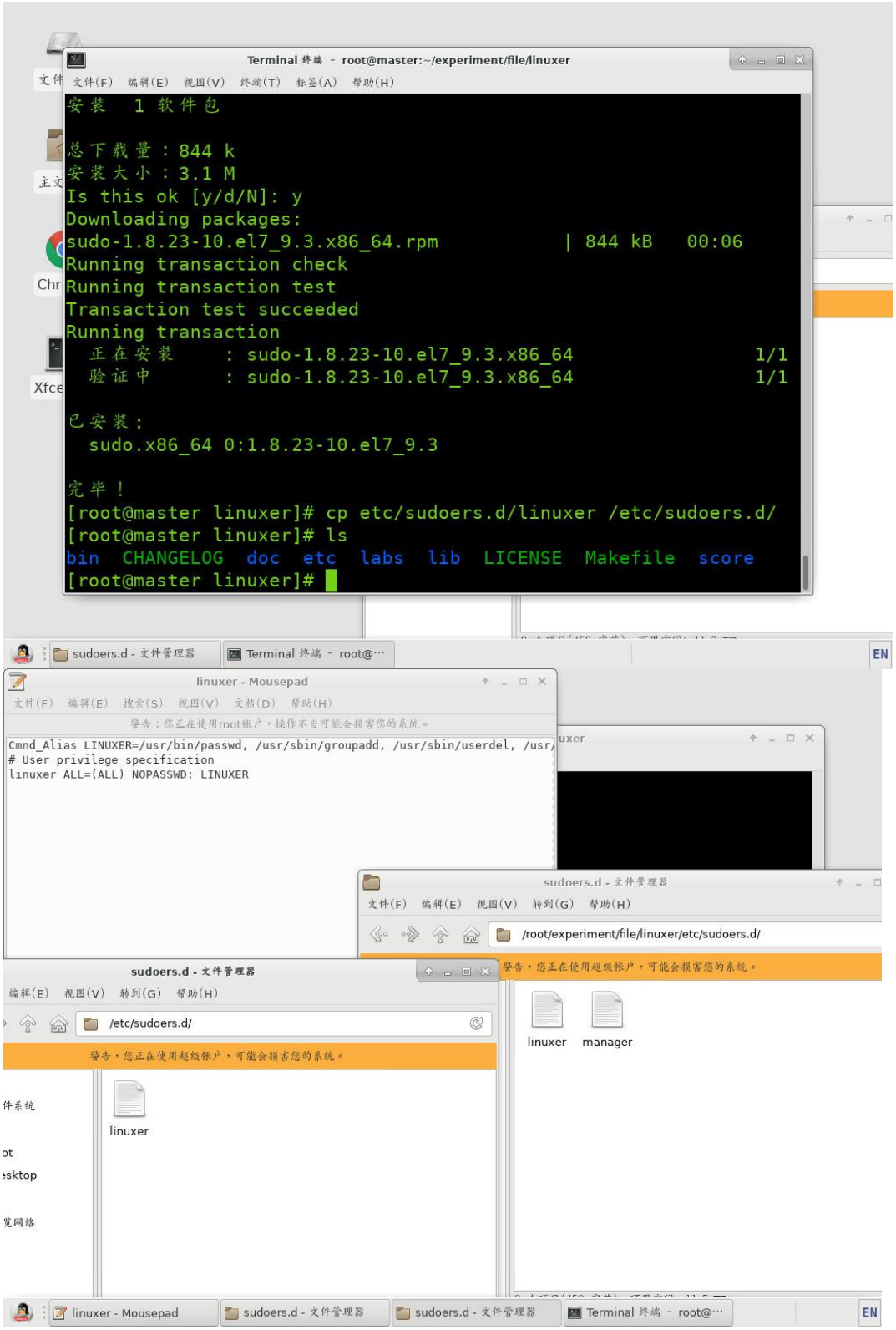
解压 linuxer，./configure 进行编译，make 启动，安装完毕。



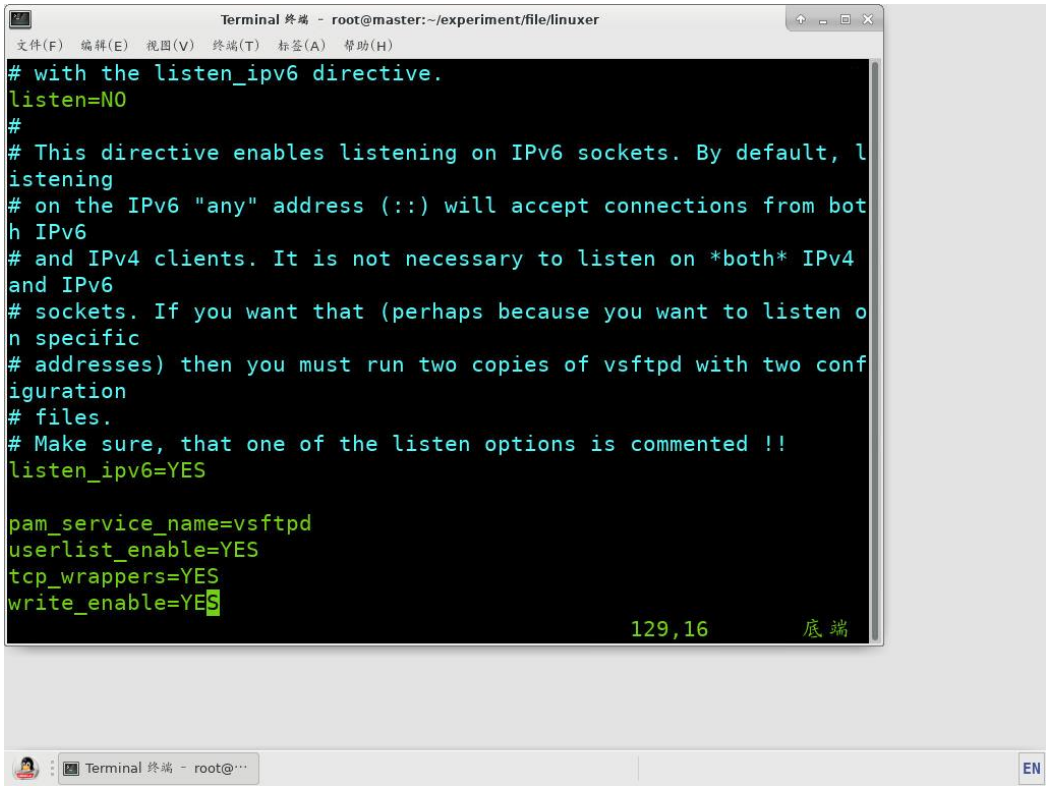
创建服务帐号 linuxer 和教师账号 teacher，并通过设置这两个帐号的登陆 shell 让客户端远程连接自动运行服务器代码。口令设置为 linuxer。



拷贝实验系统目录 cp etc/sudoers.d/linuxer /etc/sudoers.d/



将 linuxer 写入权限打开，输入 vim /etc/vsftpd/vsftp.conf 最后一行写入 write_enable=YES



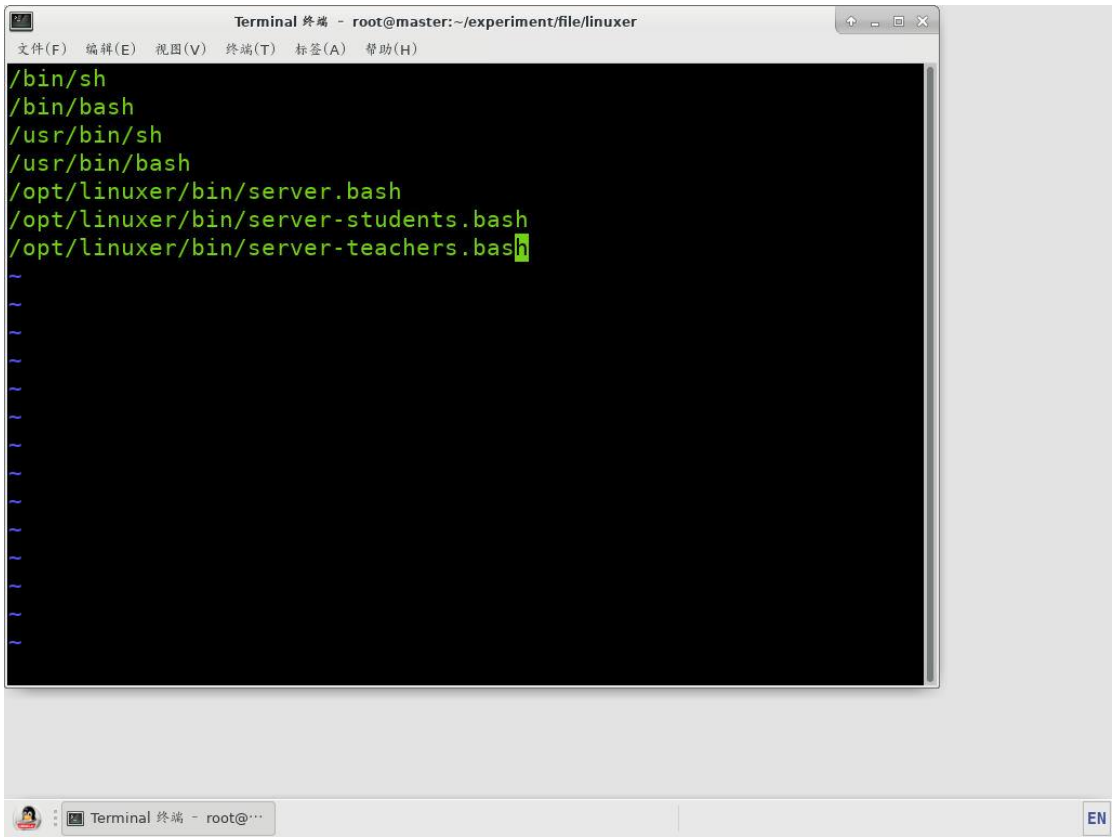
在 vim /etc/shells/中添加

/opt/linuxer/bin/server.bash

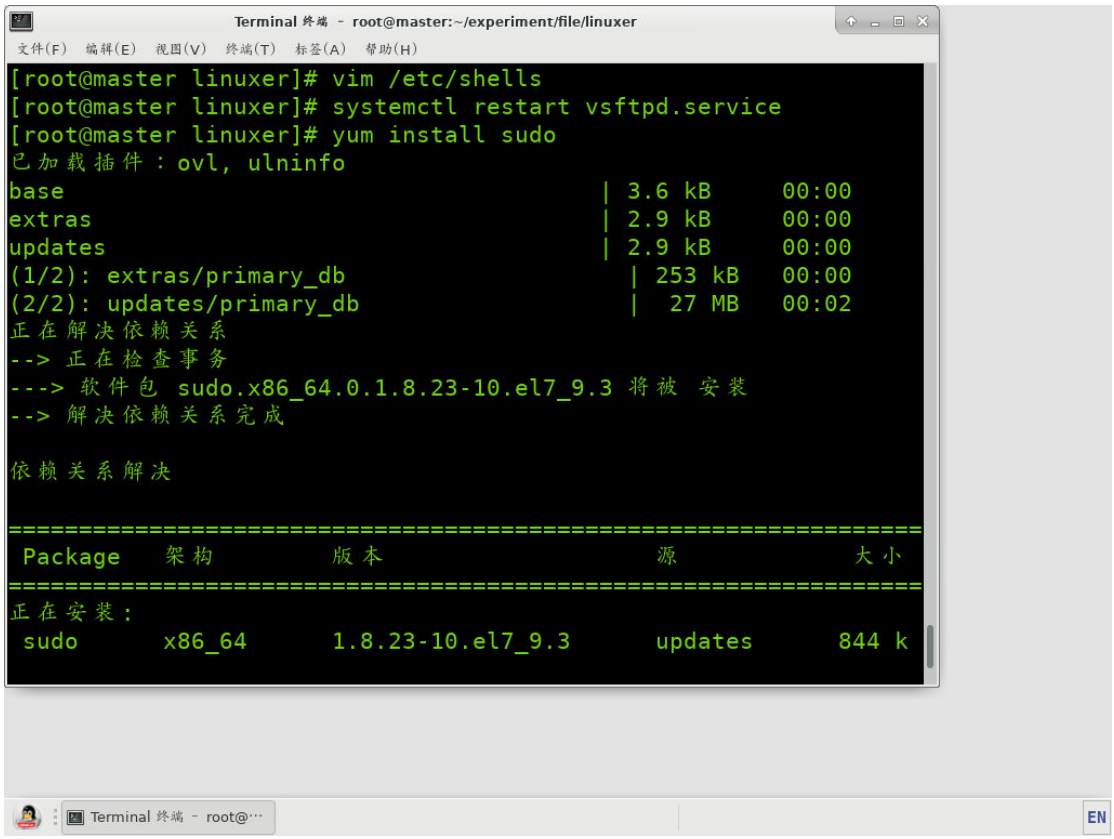
/opt/linuxer/bin/server-students.bash

/opt/linuxer/bin/server-teachers.bash

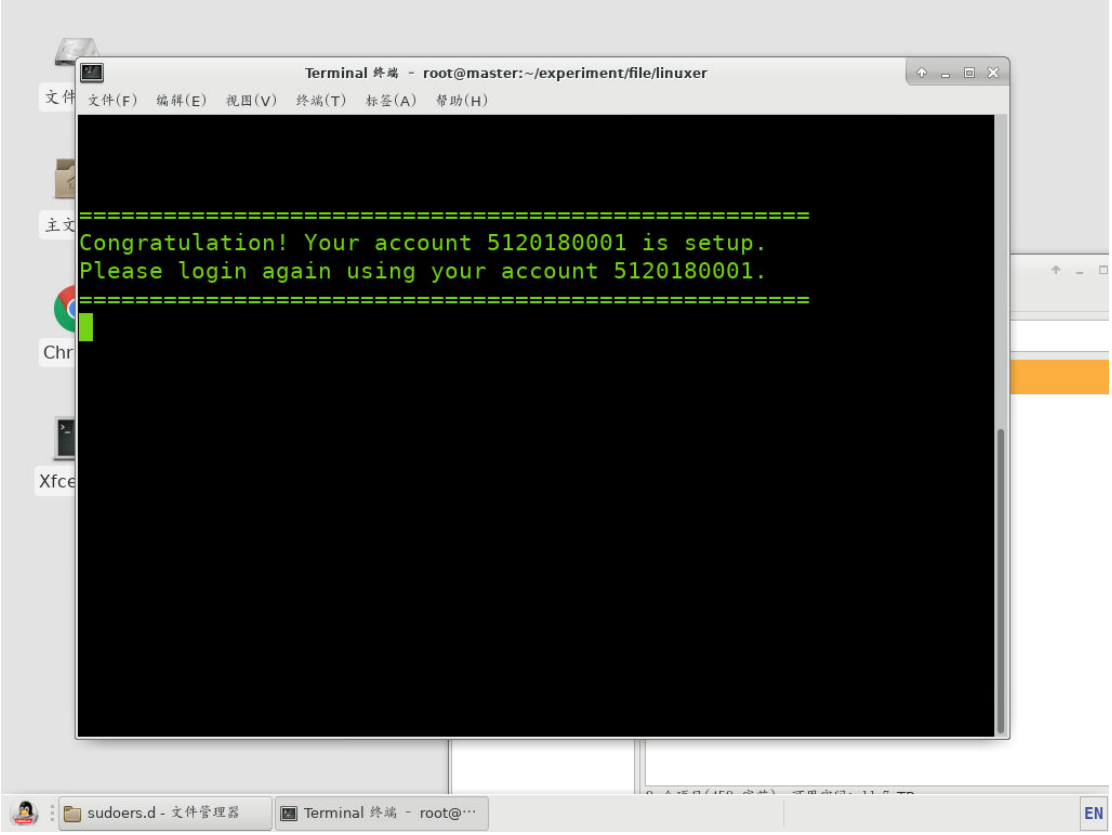
指定这些程序作为系统用户的登录 shell。



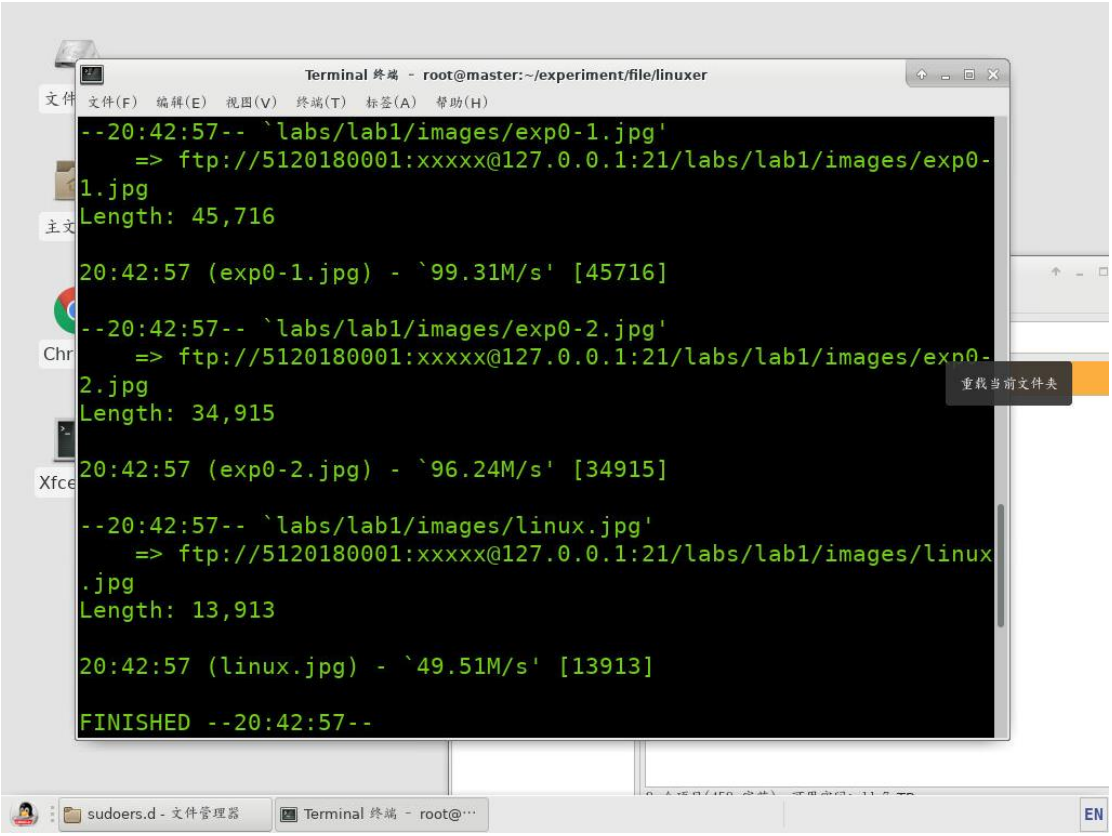
重启服务生效，提升权限下载 yum install sudo



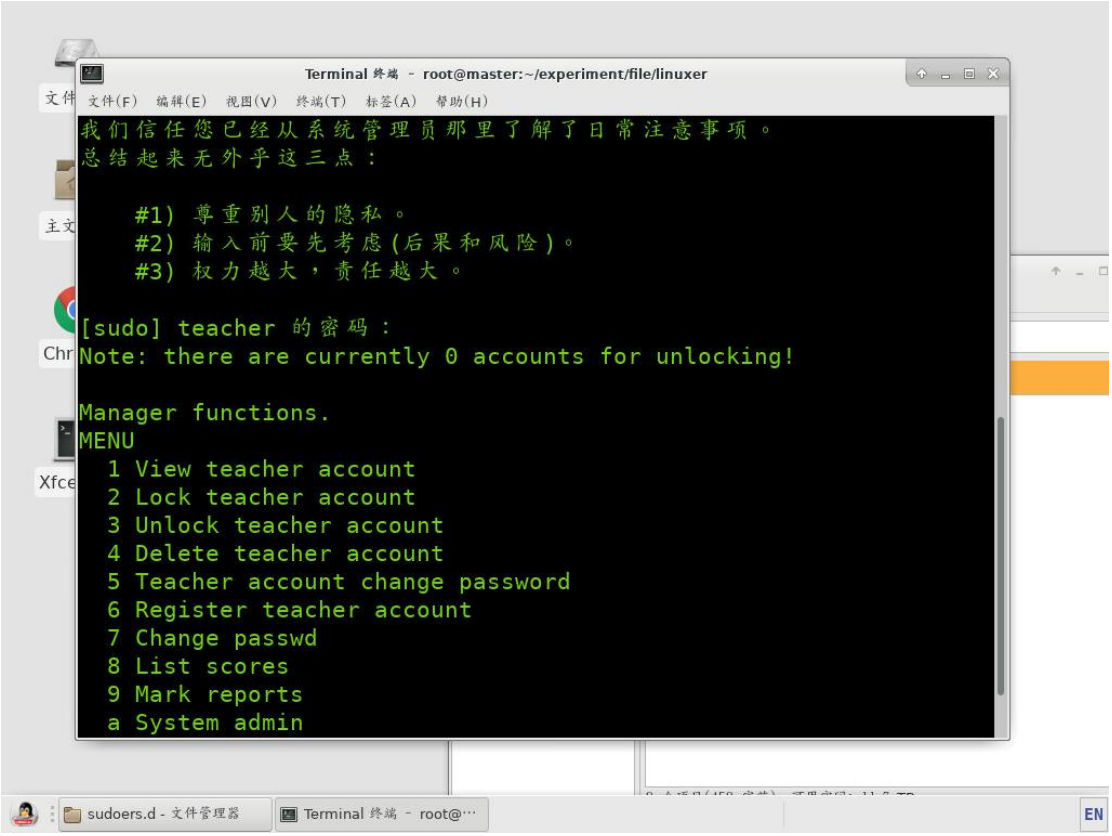
进入 linuxer 创建学生账号 5120180001



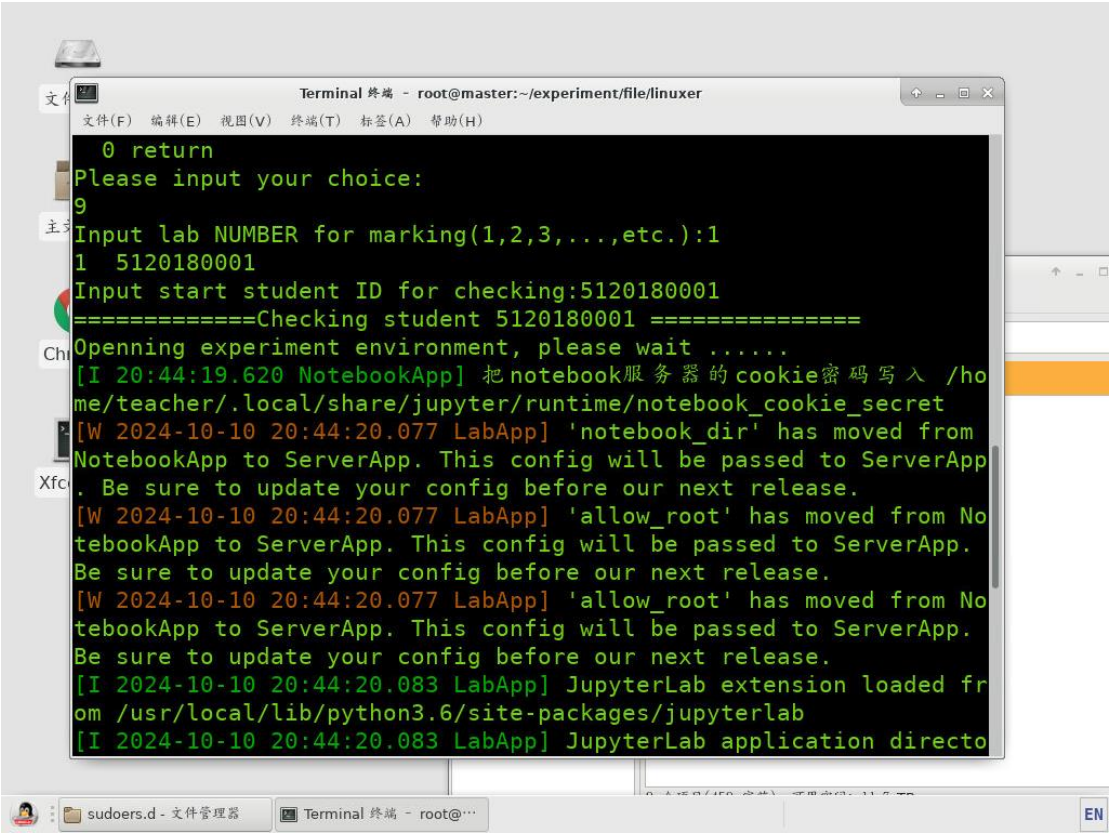
上传实验报告


Terminal 终端 - root@master:~/experiment/file/linuxer
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
--20:42:57-- `labs/lab1/images/exp0-1.jpg'
=> ftp://5120180001:xxxxx@127.0.0.1:21/labs/lab1/images/exp0-1.jpg
Length: 45,716
20:42:57 (exp0-1.jpg) - `99.31M/s' [45716]
--20:42:57-- `labs/lab1/images/exp0-2.jpg'
=> ftp://5120180001:xxxxx@127.0.0.1:21/labs/lab1/images/exp0-2.jpg
Length: 34,915
20:42:57 (exp0-2.jpg) - `96.24M/s' [34915]
--20:42:57-- `labs/lab1/images/linux.jpg'
=> ftp://5120180001:xxxxx@127.0.0.1:21/labs/lab1/images/linux.jpg
Length: 13,913
20:42:57 (linux.jpg) - `49.51M/s' [13913]
FINISHED --20:42:57--
重载当前文件夹
sudoers.d - 文件管理器
Terminal 终端 - root@…
EN

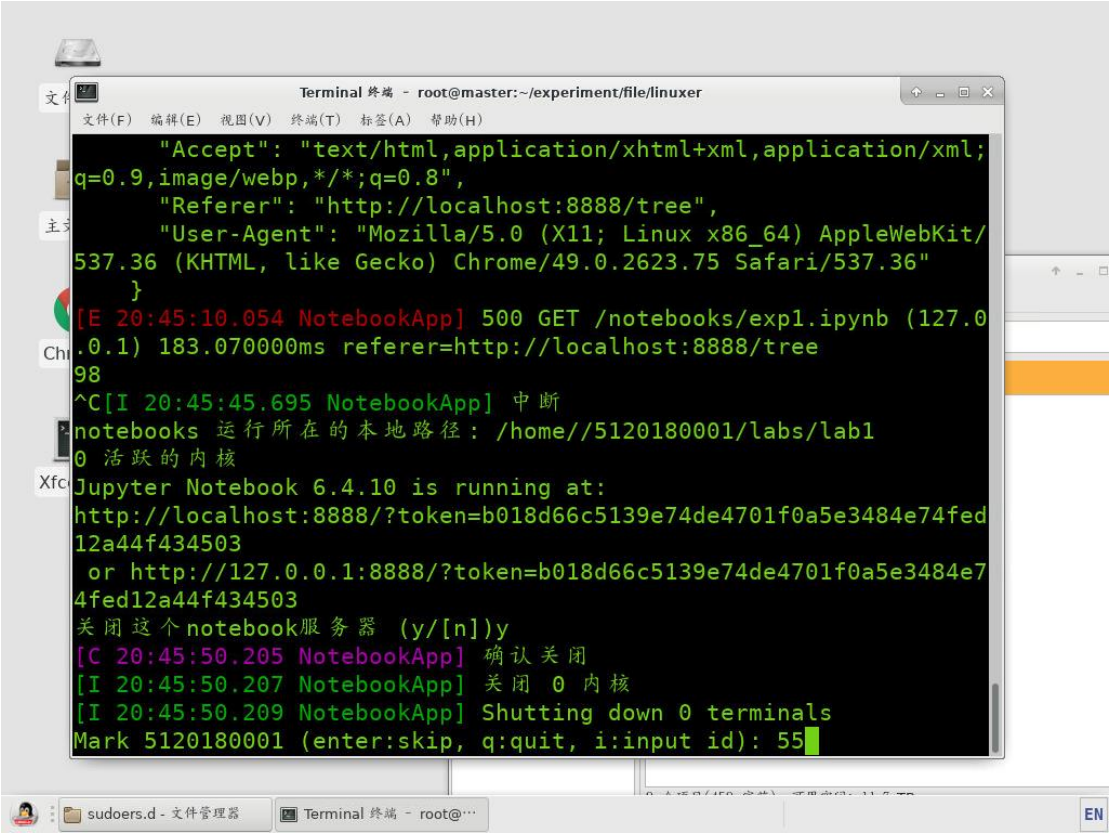
进入教师账号


Terminal 终端 - root@master:~/experiment/file/linuxer
文件(F) 编辑(E) 视图(V) 终端(T) 标签(A) 帮助(H)
我们信任您已经从系统管理员那里了解了日常注意事项。
总结起来无外乎这三点：
#1) 尊重别人的隐私。
#2) 输入前要先考虑(后果和风险)。
#3) 权力越大，责任越大。
[sudo] teacher 的密码：
Note: there are currently 0 accounts for unlocking!
Manager functions.
MENU
1 View teacher account
2 Lock teacher account
3 Unlock teacher account
4 Delete teacher account
5 Teacher account change password
6 Register teacher account
7 Change passwd
8 List scores
9 Mark reports
a System admin
sudoers.d - 文件管理器
Terminal 终端 - root@…
EN

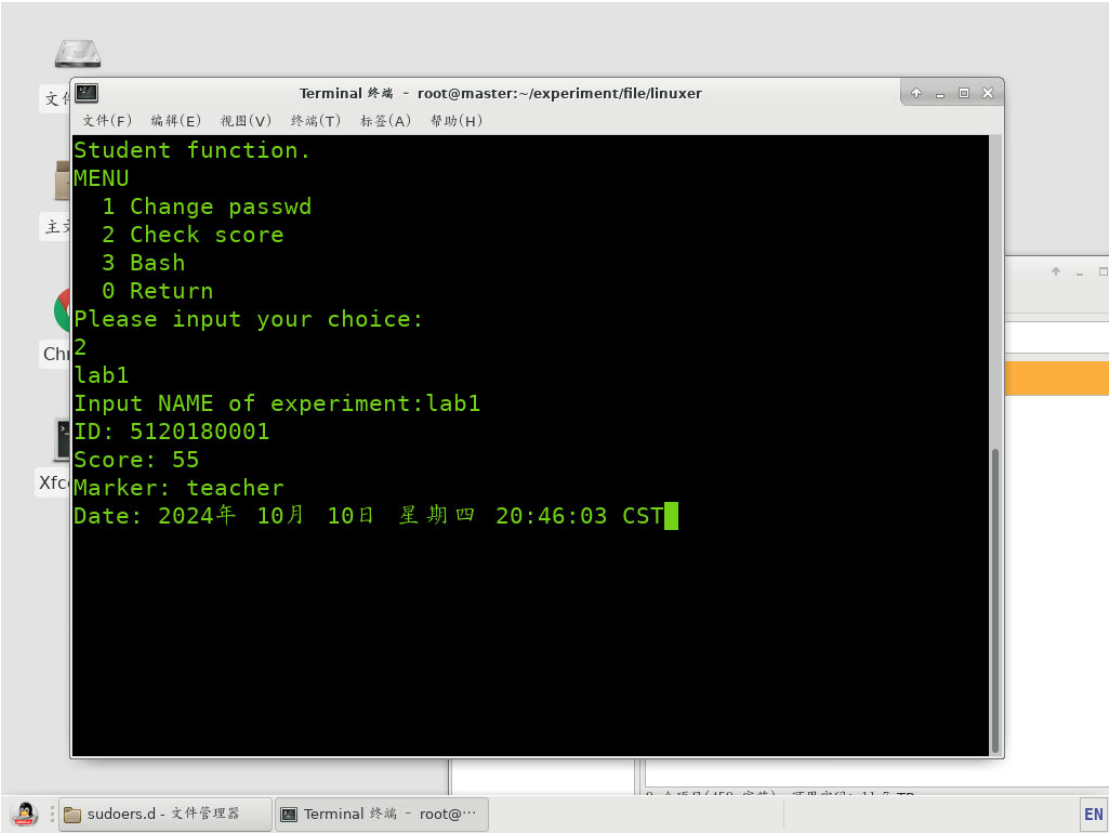
批改报告



给一个 55 分



退出教师账号返回学生账号查看成绩



四、实验思考或体会

在本次实验中，我通过配置 Linux 系统服务器端来掌握服务配置的基本方法，这个过程不仅加深了我对 Linux 系统管理的理解，还提升了我在实际操作中配置和管理服务的能力。

我创建了服务账号`linuxer`和教师账号`teacher`，并通过设置这两个账号的登录 shell 来自动运行服务端代码。这是通过修改`/etc/passwd`文件来实现的，确保了当用户远程连接时，系统会自动启动指定的服务脚本。

我通过修改`/etc/sudoers.d/linuxer`文件，赋予了`linuxer`账号必要的 sudo 权限。同时，我还调整了 FTP 服务器配置文件`/etc/vsftpd/vsftp.conf`，开启了写入权限，允许用户上传和下载文件。

此外，我还学习了如何通过`/etc/shells`文件来指定用户的登录 shell，这对于控制用户登录后的行为非常有帮助。通过添加特定的服务脚本路径，我可以确保用户登录后自动运行特定的环境设置或应用程序。

我通过设置教师和学生账号的登录权限，实现了学生上传实验报告和教师批阅的流程。这个过程不仅锻炼了我的系统配置能力，还让我对 Linux 环境下的用户管理和权限控制有了更深入的认识。