CS110 Discussion

Introduction of Linux

Yuxuan LI

VSPLab@SIST

2026.03.06

liyx22025@shanghaitech.edu.cn

目录

1 Linux 是什么

当我们谈及 Linux 时，我们到底在说什么?

“我只想插一句话，你所说的 Linux，实际上是指 GNU/Linux。”[1]

[1] https://www.gnu.org/gnu/linux-and-gnu.html

操作系统：一段简史

Unix 是第一个现代意义上的操作系统，它诞生于 20 世纪 70 年代的 AT&T 贝尔实验室。在最初的岁月中，Unix 是相当开放的，以非常低廉的价格授权给学术机构使用。然而到 20 世纪 80 年代，AT&T 被反垄断法限制，但逐渐意识到了 Unix 系统的商业价值，开始收紧使用许可，使 Unix 系统成为昂贵的商业软件，自由不再。

操作系统：一段简史

历史上，为了在 Unix 之外重建自由世界，曾有两条不同技术路线的探索

- BSD: Berkeley Software Distribution.
- GNU: GNU is not Unix.

BSD：Unix 的继承者

BSD 的全称是 Berkeley Software Distribution。BSD 来自伯克利对 Unix 原始代码的改进与增强，最初是包含 Unix 专用代码的。然而，当 AT&T 开始维权时，BSD 不得不重写了所有可追溯到 AT&T 的代码，净化出了一个干净且无法律风险的版本。

BSD 可以称之为一个纯正的 Unix 系统，它是 Unix 系统的直接后裔，今天比较流行的版本有 FreeBSD、OpenBSD、NetBSD。相当遗憾的是，尽管 BSD 最终赢得了诉讼，但却错失了最重要的发展机遇，使 GNU/Linux 后来居上。不过，由于 BSD 的开源许可非常宽松，允许商业软件使用其代码并保持闭源，因此，当今三大操作系统之一的 MacOS 事实上就是基于 BSD 的系统！故 MacOS 也是一种 Unix 系统。

GNU：Unix 的重构者

GNU 的全称是 GNU is not Unix，许多开源项目偏爱这类颇具趣味的递归缩写。

正如它的名称所言，GNU 完全没有使用 Unix 系统的任何代码，目标是参照 Unix 系统的功能，独立开发一套兼容 Unix 的自由操作系统。可以说，GNU 几乎代表了自由软件的精神，而且从一开始，GNU 计划的目标就是实现一个完整的操作系统！

事实上，他们也几乎成功了！GNU 实现了系统所需的所有用户空间工具（或许你听说过 bash 和 gcc），却唯独缺少一个可用的内核。GNU 的内核项目称为 Hurd，由于技术路线的选择，Hurd 的开发进度非常缓慢，迟迟未能推出一个可用的稳定版本。

Linux 内核！从 GNU 到 GNU/Linux

而就在这个历史的关键节点上，Linux 出现了！

Linux 的全称同样是一个递归缩写 Linux is not Unix，但和 GNU 不同的是，Linux 本身并不是一个完整的操作系统，它只是一个内核，却恰巧在当时填补了 GNU 计划长期缺乏可用内核的空白。这两者的结合，最终构成了今天的 GNU/Linux 系统！

Linux 内核和 GNU 都采用了 GPL 开源协议，它要求任何使用其代码的项目都必须保持开源且继续使用 GPL 协议，这是一个非常严格且具有传染性的开源协议。

Linux 内核！从 GNU 到 GNU/Linux

因此，今天当我们谈及所谓的 Linux 系统，讨论 Linus Torvalds 在 Linux 内核上做出的贡献时，请同样不要忘记 Richard Stallman 领导的 GNU 计划为此做出的努力。

所有 Linux 系统，实际上都应该被称为 GNU/Linux 系统！

Linux 发行版

Linux 系统有许多具体的发行版，大致可以分为三个分支

- Debian 系：使用 `apt` 作为包管理器，代表发行版有 Debian、Ubuntu。
- Centos 系：使用 `yum` 作为包管理器，代表发行版有 Fedora、Centos、Redhat。
- Arch 系：使用 `pacman` 作为包管理器，代表发行版有 Arch Linux。

在本课程中，推荐使用 Ubuntu 24.04，这是其最新的长期支持版（LTS）。

Windows 是一个糟糕的系统，尤其对于开发者而言！

Windows 是今天最为流行的操作系统，但它却是一个格格不入的系统！

- 路径使用反斜杠 \ 作为分割符，但反斜杠又是常用的转义符。
- 路径是以盘符 `C:` 起始，没有根目录的概念。
- 文件名对大小写不敏感，例如 `image.png` 和 `Image.PNG` 是同一个文件。
- 没有标准的 `/bin` 和 `/lib` 目录，软件被安装在 `C:\Program Files` 下。
- 没有包管理器的概念，软件需要通过浏览器下载并手动安装。

究其原因，是因为 Windows 是当今三大操作系统中唯一和 Unix 没有继承关系的系统，它不遵守大部分类 Unix 系统共同遵循的 POSIX 规范！同时，系统中也充斥着许多糟糕和臃肿的设计。这导致任何在 Windows 上的开发工作会变得格外困难！

Linux 使用的几点提示

- Linux 不止是命令行，但也不要惧怕命令行！
- Linux 是一个健全的操作系统，你甚至可以玩 Minecraft！
- 你不必使用很多软件，但出于各种原因，有时这可能并不容易。

Linux 安装的引导

推荐使用双系统的方式安装 Linux 系统，不鼓励使用虚拟机

- 下载 Ubuntu 24.04 的系统镜像[2]（ISO）。
- 下载 BalenaEtcher[3]或 Rufus[4]，这是一个启动盘制作工具。
- 准备一个 U 盘，将系统镜像通过启动盘制作工具烧入 U 盘。
- 若 Windows 启用了 Bitlocker，**请务必先关闭，否则会导致 Windows 损坏**！
- 在 Windows 的磁盘管理中，压缩卷，留出未分配空间用于安装 Ubuntu 系统。
- 将启动盘插入，重启电脑，按键（F2、F10、DEL）进入 BIOS，选择从 U 盘启动。

[2] https://ubuntu.com/download/desktop?version=24.04
[3] https://etcher.balena.io
[4] https://rufus.ie

Linux 安装的引导

从 U 盘启动后，会有图形界面引导完成 Ubuntu 系统的安装，有几点要注意

- 选择系统语言中文，以便自动安装中文输入法和中文字体。
- 选择安装第三方驱动，以便自动安装 NVIDIA 显卡驱动。
- 在分区配置中，选择手动分区，选中先前留出的未分配空间创建分区，分区的文件系统推荐选择 Btrfs 而不是默认的 Ext4，分区应挂载于 / 下。创建分区会格式化对应区域，因此，**请务必不要错误的格式化 Windows 正在使用的区域**!

2 Linux 的基本命令

软件包管理器

在 Windows 系统上，软件的下载、安装、更新是一个相当无序的过程：软件的安装程序需要通过浏览器从不同网站下载，安装程序需要手动运行，安装路径也非常随意。软件的更新则更是混乱，有些完全不支持更新，有些则会后台静默的热更新。

软件包管理器

在 Linux 系统中，几乎所有发行版都采用了某种软件包管理器，例如，Ubuntu 的软件包管理器就是 apt，你可以用 apt 命令完成大部分软件的下载、安装、更新！

命令 apt

通过 apt 可以进行更新，更新会分为两步

```
1 sudo apt update
2 sudo apt upgrade
```

其中，update 的作用是刷新软件包索引，以了解有哪些软件可以更新，upgrade 的作用是真正更新软件包，sudo 的含义是以 root 身份运行。在 Linux 系统中，系统软件和应用软件是一体的，安装完系统后，应当立即运行上述两条命令进行更新。

使用 sudo 需输入密码。在 Linux 下输入密码是没有回显的，盲输按回车即可。

命令 apt

通过 apt 可以安装软件，例如以下命令会安装 vim，这是一个命令行上的编辑器

```
1 sudo apt install vim
```

你不需要指定安装路径！这在 Linux 系统下有一套严格的规范。

命令 pwd

pwd 用于显示当前路径（Print Working Directory）

```
1 pwd
```

你可能会得到类似 /home/liyuxuan 的结果，事实上，当前路径总是会显示在命令提示符上，只不过当前用户的家目录会用 ~ 表示，终端启动时默认位于家目录。

命令 ls

ls 用于显示当前路径下的文件（List Directory Contents）

```
ls
```

搭配 -l 显示详细信息，搭配 -a 显示隐藏文件，也可以查看指定路径的文件

```
ls -l
ls -a
ls /home
```

Linux 命令中，参数以空格分割，选项以 - 或 -- 开头（短选项/长选项）。

命令 cd

cd 用于移动当前路径（Change Directory）

```
1 cd /home
```

回到家目录，你可以尝试不同的路径表示方法

```
1 cd /home/liyuxuan
2 cd liyuxuan
```

Linux 中，若路径以根目录 / 开头，那就是绝对路径，否则是相对路径。

命令 cd

有两个特殊的目录：用 . 代表当前目录，用 .. 代表上一级目录

```
cd .
cd ..
```

Linux 中，所有以 . 开头的文件都是隐藏文件，用 `ls -a` 查看。

命令 touch

`touch` 用于刷新时间戳，但如果目标文件不存在，它会创建一个空文件

```
1 touch file.txt
```

你可以用 `ls -l` 查看文件的时间戳（即最后修改时间）。

命令 mkdir

`mkdir` 用于创建目录（Make Directory）

```
1 mkdir dir
```

你可以在 `ls -l` 中观察第一列，若以 `d` 开头则为目录，若以 `-` 开头则为普通文件。

命令 echo

echo 用于打印文字，请注意任何包含空格的参数需要用双引号包裹

```
1 echo "Hello World"
```

通过 > 和 >> 可以将命令行上的输出重定向到文件，> 是覆盖，>> 是追加

```
1 echo "Hello World" > file.txt
2 echo "Hi!" >> file.txt
```

命令 cat

cat 用于查看文件内容（Concatenate）

```
1 cat file.txt
```

该命令实际上是 Concatenate 的缩写，但不妨理解为一只猫挠了一下文件（^_^）。

命令 vim

vim 是一个命令行上的代码编辑器（你可以用它来编辑任何文本文件!）

```
1 vim main.c
```

Vim 有一套非常高效的快捷键体系，但它的使用方式常常会让新手感到困惑，牢记以下几点可以让你至少把它当成一个普通编辑器来使用。Vim 在启动默认处于普通模式，此时，键盘上几乎每一个按键都是快捷键！你会发现你无法编辑文件，正确的做法是按下 I，这会让你进入插入模式，这就和普通编辑器一致了。完成后，你需要按 ESC 回到普通模式，输入 :wq 进行保存和退出（记住这一点尤为重要[5]）。

[5]关于 Vim 一个经久不衰的笑话是：如何退出 Vim?

命令 mv

mv 用于重命名或移动文件（Move File）

```
1 mv file.txt file.md
2 mv file.txt dir
3 mv file.txt dir/file.md
```

- 若目标是一个已经存在的目录，文件会被移动到该目录下。
- 若目标不存在或不是目录，文件会被重命名并移动至目标指定的路径。

mv 不区分操作对象是文件还是目录，也可以用来重命名或移动目录。

命令 cp

cp 用于复制文件（Copy File）

```
cp main.c main.cpp
cp main.c dir
cp main.c dir/main.cpp
```

搭配 -r 选项可以递归复制目录

```
cp -r dir source
```

命令 rm

`rm` 用于删除文件（Remove File）

```
1 rm main.c
```

搭配 `-r` 选项可以删除目录

```
1 rm -r dir
```

特别需要强调的是，Linux 的删除是没有回收站的，三思而后行！

按下回车前请三思

命令行非常高效且强大，但错误的使用会造成格外严重的后果！

- 我正在操作的文件是我想要操作的吗？
- 我是否位于正确的目录？这是在本地还是在服务器？
- 我是否真的理解我在做什么？尤其是当这条命令是由 AI 提供的。

胆大心细，勇敢的探索和学习，谨慎的实施和执行，对自己的行为负责。

著名的危险命令

以下是一条非常著名的危险命令，**绝对不要在任何情况下尝试！**

```
1 sudo rm -rf /
```

其中，`sudo` 用于提权，`rm` 用于删除，`-r` 允许递归删除，`-f` 允许强制删除，这会跳过部分带保护的文件的删除确认，最后，删除对象是 `/` 即根目录，因此这条命令会高效且悄无声息的抹去这台 Linux 系统上的一切数据，甚至包括操作系统自身！

能力越大，责任越大，请谨慎对待你手中掌握的权限。

命令 ln

`ln` 用于创建链接（Link File），选项 `-s` 代表创建的是软链接

```
1 ln -s dir link
```

软链接的工作方式类似于快捷方式，当访问 `link` 时，其实是在访问 `dir` 目录，但和快捷方式不同的是，你的路径不会跳转，你仍然会位于 `/home/liyuxuan/link` 下。

软链接在 `ls -l` 下会显示为 `l` 类型，代表这是一个链接。

软链接与硬链接

链接分为两种，软链接（由 `ln -s` 创建）和硬链接（由 `ln` 创建）

- 软链接类似于 C++ 语言中的指针（本质上是存储地址的变量）。
- 硬链接类似于 C++ 语言中的引用（本质上是一个别名）。

在文件系统层面，文件名和文件本体（inode）通常是一一对应的。正所谓引用即别名，硬链接不会创建新的文件，而是令一个文件 inode 能关联到多个文件名上。换言之，如果将硬链接理解为文件名和 inode 间的链接关系，那么所有文件其实都是硬链接：有些 inode 只关联到一个文件名，有些 inode 会关联到多个文件名。硬链接命令的作用是在 inode 和文件名之间创建额外的链接。当删除文件时，本质上是在减少 inode 的链接计数，当 inode 的链接计数归零时，对应空间才真正被释放。

软链接与硬链接

软链接则是完全不同的东西，它是一个独立的文件（有自己的 inode），只不过这个特殊的文件存储的内容是一个路径，使得你在访问它时实际访问到的是别的地方。

- 软链接可以链接目录或文件，硬链接通常只被允许链接文件。
- 软链接可以跨文件系统链接，硬链接则不可以（inode 对每个文件系统独立）。
- 软链接只是一个普通文件，删除软链接无影响，删除源则所有软链接失效。

最佳实践：除非你很清楚自己要做什么，**总是使用软链接**！

获取帮助

几乎所有的命令都可以通过 `--help` 选项查看帮助

```
1 ls --help
```

除此之外，使用 `man` 可以查看某个命令文档

```
1 man ls
```

命令补全

你不必完整输入命令！下面展示了输入命令时如何通过按 Tab 来自动补全

```
1 mkdi<Tab>
2 mkdir /h<Tab>
3 mkdir /home/li<Tab>
4 mkdir /home/liyuxuan/vsp
```

简而言之，按 Tab 可以帮助你补全命令和已经存在的路径。但补全不是魔法，它不能预知你的想法，例如这里 `vsp` 肯定无法补全出来，因为这是你将要创建的目录。

命令历史

你不必重复输入已经输入过的命令！按向上键和向下键可以翻阅输入过的历史命令。

除此之外，使用 `history` 可以查看所有输入过的命令

```
1 history
```

如果你要搜索包含特定内容的历史命令，可以用管道 | 连接 `grep`

```
1 history | grep mkdir
```

管道 | 用于将一个命令的输出重定向到一个命令的输入，`grep` 用于文本匹配。

命令 tar

在 Linux 中，经常会看到 .tar.gz 后缀的压缩包

- .tar 代表归档，即将多个文件打包为单个文件，由 tar 命令完成。
- .gz 代表压缩，即将单个文件进行压缩，由 gz 命令完成。

但事实上，通过恰当的选项，可以用 tar 命令一步完成 .tar.gz 的压缩和解压缩。

命令 tar

tar 用于压缩的命令是

```
1 tar -czvf package.tar.gz dir
```

tar 用于解压的命令是

```
1 tar -xzvf package.tar.gz
```

其中，选项 -c 代表 Create，选项 -x 代表 Extract，选项 -z 代表使用 Gzip 格式的压缩算法，选项 -v 代表 Verbose，这会列出所有压缩和解压的文件（需要静默输出时可以省略），选项 -f 代表 File，即指定操作的归档文件。这些选项中 -f 必须放在最后，因为其后要跟随一个参数。另外，短选项允许连写是一个普遍支持的特性。

3 Linux 的目录结构

Linux 的目录结构

Linux 系统的目录结构遵循 Filesystem Hierarchy Standard（FHS）。事实上，这几乎是所有类 Unix 系统遵循的规范，你甚至可以在 MacOS 上找到相似的目录结构！

表 1: Linux 的根目录结构

目录	用途	目录	用途
/bin	用户执行程序	/media	自动挂载点
/lib	动态与静态库	/mnt	手动挂载点
/sbin	系统执行程序	/boot	启动引导
/home	普通用户家目录	/dev	设备文件
/root	超级用户家目录	/sys	设备属性
/etc	配置文件	/proc	进程信息
/var	可变文件	/run	进程数据
/usr	系统资源	/tmp	临时数据
/opt	可选软件	/srv	服务数据

目录 /bin /lib /sbin

/bin 目录存放了所有用户可以执行的命令

```
1 ls /bin | grep ls
2 ls /bin | grep mkdir
```

/bin 目录也包含大部分通过 apt 安装的程序。对于 Linux 而言，它并不区分“系统自带的命令”和“用户安装的应用程序”，它们本质上都是可执行的二进制程序。

目录 /bin /lib /sbin

例如，安装 build-essential 包，这包含了 C/C++ 开发的基本工具

```
sudo apt install build-essential
```

你现在就可以在 /bin 下找到 gcc 和 make 等工具了

```
ls /bin | grep gcc
ls /bin | grep make
```

目录 /bin /lib /sbin

/lib 目录存放了所有动态库和静态库

- 库的本质是预先编译好的通用二进制片段，它可以和应用程序进行链接。
- 动态链接库：运行时链接，后缀为 .so，例如 C 库 libc.so
- 静态链接库：编译时链接，后缀为 .a，例如 C 库 libc.a

你可以在 /lib/x86_64-linux-gnu 下找到 libc.so 和 libc.a

```
1 ls /lib/x86_64-linux-gnu | grep libc.so
2 ls /lib/x86_64-linux-gnu | grep libc.a
```

目录 /bin /lib /sbin

通常来说，动态链接是较为常用的。当你在 C 程序中写下 `#include <stdio.h>` 以及 `printf` 并编译时，函数 `printf` 的二进制并不会被链接到你的程序中，真正发生的事情是，当你的程序运行到 `printf` 时，系统会找到 `libc.so` 并运行对应的二进制，这就是所谓的“动态链接”即“运行时链接”。动态链接机制使系统中海量的二进制程序不必重复包含 C 库的片段，而是可以共享一个 C 库，从而节约大量空间！

然而，静态链接在某些情况下也有其意义。当你的 C 程序最终是在某个裸机的单片机上运行时，根本没有操作系统进行动态链接，此时就需要在编译阶段静态链接用到的函数。这从空间上也很划算，你只需要带上 C 库中你用到那一部分就可以了！

目录 /bin /lib /sbin

/sbin 目录存放了所有 root 才能执行的命令

```
1 ls /sbin | grep poweroff
2 ls /sbin | grep reboot
```

这是因为，在多人使用的服务器上，普通用户显然不能随意关机和重启！

目录 /home /root

/home 和 root 都是用户的家目录

- /home 是普通用户的家目录，每个用户在 /home 下有同名的家目录。
- 用户 liyuxuan 的家目录位于 /home/liyuxuan。
- /root 是超级用户的家目录，这是 root 作为超级用户在架构上的特权，其家目录并非位于 /home/root，而是高挂了一级，直接位于根目录 /root 下。
- 用户的家目录始终可以用 ~ 表示（取决于当前登录的用户）。

目录 /etc

/etc 下存放配置文件，比较典型的例子是 hosts 文件

```
1 sudo vim /etc/hosts
```

你可以在这里设置本地的 DNS 解析，例如给你的课程服务器起一个别名

```
1 10.15.28.28 ee219
2 10.15.27.77 ee291f
```

目录 /var

/var 下存放可变文件，比较典型的例子是 apt 的日志

```
1 cat /var/log/apt/history.log
2 cat /var/log/apt/term.log
```

你可以在两个文件中分别看到 apt 最近的更新记录和当时的命令行输出内容。

目录 /usr

/usr 的全称是 Unix System Resource，请注意其和 User 无关！

如果你足够细心，仔细观察了 `ls -l /` 的输出，你会注意到 /bin /lib /sbin 实际上分别是指向 /usr/bin /usr/lib /usr/sbin 的软链接！这是因为早期的实践中，前者存储系统所需的命令和库，后者存储用户安装的命令和库，但是随着时代发展，现在主流的 Linux 发行版都选择将这两个目录用软链接的方式合并在一起。

目录 /opt

/opt 用于安装可选的第三方大型软件，例如

- Wolfram Mathematica
- Xilinx Vivado
- 腾讯会议

这些软件不遵守标准的目录结构，因此统一在 /opt 下为每个软件分配独立的目录。

目录 /media /mnt

/media 和 /mnt 分别是自动挂载点和手动挂载点，例如当插入 U 盘后

```
1 ls /media/liyuxuan
```

假如 U 盘的名称是 udisk，那你就会在该位置看到一个名为 udisk 的目录，该过程称为挂载（Mount），这和 Windows 中插入 U 盘会出现新的盘符的行为是不一样的。

通常移动存储设备都可以被自动挂载至 /media，极少数情况需要手动挂载至 /mnt。

目录 /boot

/boot 目录包含了操作系统正常启动所必须的文件。事实上，在安装系统进行分区时，除了挂载于 / 的主分区，还会自动产生一个挂载于 /boot/efi 的启动分区。

目录 /dev

/dev 用于存放设备文件，例如

```
1 ls -l /dev | grep nvme
2 df -h
```

第一条命令的输出可能会看到 nvme0n1p1 和 nvme0n1p2，这对应了 NVMe 上连接的 SSD 的第一个分区和第二个分区。第二条命令用于查看磁盘占用，对照可以发现，挂载点 / 对应 /dev/nvme0n1p1，挂载点 /boot/efi 对应 /dev/nvme0n1p2。

目录 /dev

在 Linux 系统中，一切皆文件。磁盘设备本身也被视为一个文件，这并不是一个真实存在的文件，而是通过文件读写的方式包装了对磁盘设备的数据写入和读取。挂载的作用，就是将磁盘设备的直接读写转换为所看到的文件的创建、修改、删除。

设备文件在 `ls -l` 下会显示为 `b` 或 `c` 类型，代表这是一个块设备或字符设备。

目录 /sys

/sys 用于配置设备属性，例如

```
1 echo 0 | sudo tee /sys/class/leds/asus::kbd_backlight/brightness
2 echo 1 | sudo tee /sys/class/leds/asus::kbd_backlight/brightness
```

第一条命令将关闭键盘灯，第二条命令将开启键盘灯。这里 tee 最终发挥的作用就是将 0 和 1 写入 brightness 文件（重定向 > 在这里会有权限问题）。在 /sys 下的文件是设备属性的反映，你可以通过对 /sys 下文件的读写查看和控制设备属性。

目录 /proc

/proc 是进程的映射，例如

```
1 ls /proc
2 ps -A
```

第一条命令会看到大量数字命名的目录，例如 248 3533 55991。第二条命令用于查看全部进程。对照发现，进程号和 /proc 下的数字命名的目录是一一对应的!

目录 /run /tmp /srv

/run 用于存放进程的数据，/tmp 用于存放任何临时文件，系统重启后自动清空。

/srv 用于存放本机对外提供的服务的数据（适用于服务器）。

4 Linux 的权限管理

超级用户

Linux 系统的超级用户称为 root，通常来说，修改家目录外的文件都需要 root 权限，普通用户可以通过 sudo 临时提权以 root 身份执行命令（Superuser Do）

```
1 sudo apt install build-essential
2 sudo vim /etc/hosts
```

当使用 sudo 时，你会被要求输入自己的密码（接下来一段时间不用再次输入）。

超级用户

Linux 系统中，使用 su 命令（Superuser）可以切换至 root 用户，此时你会被要求输入 root 用户的密码。然而，大部分发行版出于安全原因都禁止直接以 root 用户登录，未设置 root 密码，因此输入任何密码都是错误的！但你可以使用下面的命令

```
1 sudo su
```

换言之，临时以超级用户身份宣布将自己提升至超级用户，只需要输入自己的密码！

超级用户

当然，这并非意味着任何用户都可以轻松登录 root 用户！因为并不是所有用户都有使用 sudo 的权限。通常来说，在一台多人使用的服务器中，只有少数作为管理员的用户才能使用 sudo 命令，给予使用 sudo 的权限等同于提供 root 用户！

超级用户

在 root 下工作是危险的。引入 sudo 机制的意义在于，让那些可以使用 root 用户的人平常仍然在自己的低权限用户下工作，仅在必要时，显式以 sudo 临时提权。

权限管理机制

Linux 系统有一套完善的权限管理机制，它可以保证多个用户同时使用，但各个用户又无法触碰他人的数据。即便在个人电脑上，理解权限的概念也是很有价值的。

权限管理机制

尝试以下命令

```
mkdir dir
touch file.txt
ln -s dir link
ls -l
```

你应当会看到以下输出

```
drwxrwxr-x 1 liyuxuan liyuxuan    0 Mar  5 10:21 dir
-rw-rw-r-- 1 liyuxuan liyuxuan    0 Mar  5 10:21 file.txt
lrwxrwxrwx 1 liyuxuan liyuxuan    3 Mar  5 10:21 link -> dir
```

文件所有权

Linux 中每个文件都有两个所有权的概念：所有者（User）、所属组（Group）

- 所有者位于 `ls -l` 输出的第三列，该例中所有者为 `liyuxuan`
- 所属组位于 `ls -l` 输出的第四列，该例中所属组为 `liyuxuan`

所属组的概念服务于一个文件需要被多个用户共享的情况。不过，每个用户都会对应一个同名且仅包含其自身的主组，组 `liyuxuan` 仅包含 `liyuxuan` 这一个用户。

默认情况下，创建文件的用户和其主组会自动成为文件的所有者和所属组。

文件权限

Linux 中的权限有三类：可读（r）、可写（w）、可执行（x）

- r：最高位（r=4），读取文件，列出目录中的文件，**可读**。
- w：中间位（w=2），修改文件，增删目录中的文件，**可写**。
- x：最低位（x=1），执行文件，进入目录，**可执行**。

例如，rwx 代表可读可写可执行（记为 7），r-- 代表只读（记为 4）。

在 Linux 中，执行文件是通过权限而非后缀区分的！例如脚本 test.py 只有在添加执行权限后才能以 ./test.py 的方式运行，否则只能用 python test.py 运行。

文件权限

Linux 中每个文件对于三种用户角色（所有者、所属组、其他人）都会定义 rwx 权限

- rw-rw-r--，所有者 rw-，所属组 rw-，其他人 r--，记为 644。
- rwxrwxr-x，所有者 rwx，所属组 rwx，其他人 r-x，记为 755。
- rwxrwxrwx，所有者 rwx，所属组 rwx，其他人 rwx，记为 777。

实际上 644、755、777 分别是文件、目录、软链接的默认权限。即默认情况下创建的文件，对所有者和所属组是可读可写，对其他人可读。目录一般应当是可以进入的，所以均增加了可执行。软链接的权限无意义，实际取决于被链接文件的权限。

文件权限

既然默认对其他人是可读的，那每个用户的隐私如何保证？答案在家目录的权限

```
1 ls -l /home
```

注意到家目录 `liyuxuan` 的权限是 `750` 而非默认的 `755`，这保护了下面的文件！

工作目录的创建尝试

尽管用户应当在自己的家目录 ~ 即 /home/liyuxuan 下工作，但实践中 ~ 下往往会产生许多杂乱的用户配置文件和用户安装软件。如果有 root 权限，也可以选择在下额外创建一个和家目录平级的 /home/workspace 作为自己专用的工作目录。

请注意，在 /home 下创建目录是需要 root 权限的

```
1 sudo mkdir workspace
```

接下来要做两件事：将目录的所有权转给自己，将目录的权限设置为 750。

命令 chown

chown 用于修改所有权（Change Ownership），冒号前后分别指定所有者和所属组

```
1 sudo chown liyuxuan:liyuxuan workspace
```

由于在执行命令前，该目录还属于 root，故需加上 sudo 进行提权。

命令 chmod

chmod 用于修改权限（Change Mode），可以直接用数字标识

```
1 chmod 750 workspace
```

chmod 的用法很灵活，例如为 test.py 添加执行权限

```
1 chmod +x test.py
```

谨慎处理权限问题！遇到权限不足时随意设置 sudo chmod 777 是不负责任的行为。

5 Linux 的远程连接

远程连接

在未来的课程中，你可能会被提供一台服务器的账号，并被提供以下信息

- IP：`10.15.28.28`
- 账号：`liyx`
- 密码：`uLu3bP9DF`

你需要通过 SSH 以命令行的方式远程登录这台服务器！

命令 ssh

ssh 用于远程登录

```
1 ssh liyx@10.15.28.28
```

当输入密码后，你就远程登录至服务器了！此时，你在键盘上敲击的任何字符都会通过 SSH 协议经网络传输到服务器上，可以像本地一样在远程服务器上执行命令！

因此可以说，若拿到了一台电脑的 SSH 权限，就等同于坐在了这台电脑屏幕前！

命令 scp

scp 用于远程复制文件（上传/下载）

```
1 scp file.txt liyx@10.15.28.28:file.txt
2 scp liyx@10.15.28.28:file.txt file.txt
```

命令 passwd

任何情况下，拿到服务器的 SSH 账号后，第一件事就是修改密码！

```
1 passwd
```

免密码登录

远程连接每次都需要输入密码，这相当麻烦！可以使用密钥实现免密登录。

第一步，生成非对称密钥，`-t rsa` 代表使用 RSA 密钥，`-b 4096` 代表生成 4096 位的密钥，`-C "my-laptop"` 代表密钥的注释，交互中对每个问题输入回车保持默认即可，这会生成公钥 `~/.ssh/id_rsa.pub` 和私钥 `~/.ssh/id_rsa` 两个文件。

```
1 ssh-keygen -t rsa -b 4096 -C "my-laptop"
```

第二步，上传公钥至服务器，这需要你最后一次输入密码

```
1 ssh-copy-id liyx@10.15.28.28
```

谢谢大家！