Linux入门 - 2023年12月14日

前言

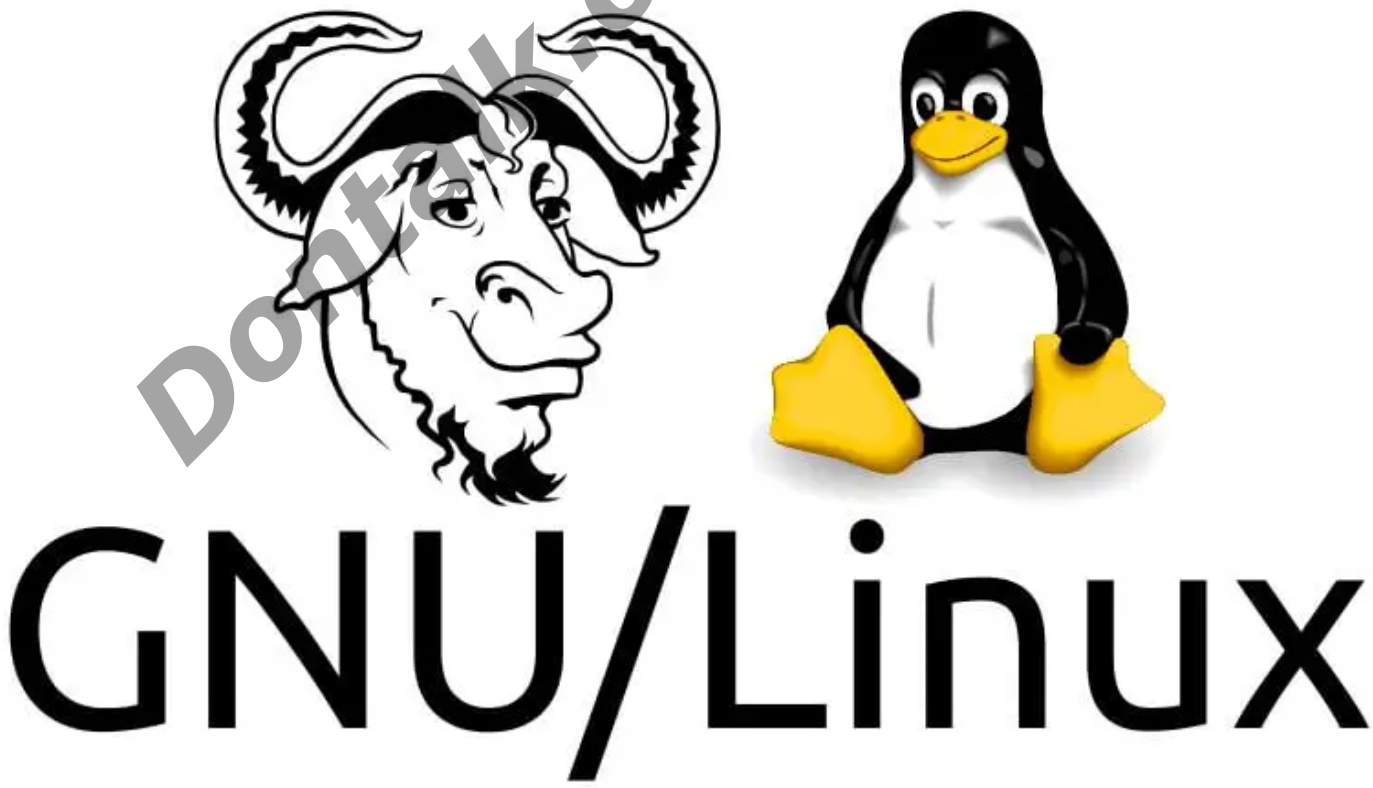
事实上，前辈MJJ已经做过「MJJ的Linux快速入门教程」来帮助各位新手MJJ快速入门Linux（但针对apt-deb系），我在这里将用更通俗简洁的语言，快速过一遍Linux，并给出apt/yum的两种不同的发行版使用入门指南。因为不是所有人都是MJJ，而这里只作快速入门，将不深入讲解Apache，Nginx，Docker，FTP，Panel等等（或许吧）

而我其实很早之前就做过Kali Linux的入门教程，但是始终觉得突兀，所以我就重新做了一个Linux的入门教程，应对我的朋友对于Linux入门可能面对的问题，该教程仅包括最基础的Linux入门以及一些历史。将不会提及Docker、FTP、Nginx、Apache、E-Mail等技术。

2023年12月14日 - **目前教程就到此结束，前前后后写了两天的教程暂时也画上了句号。希望你可以在这个教程里可以对Linux有初步的了解并入门它，由于是个人赶制，所以可能会有不少错误纰漏，还有不严谨和做得不好的地方，敬请原谅。**

Linux前世今生

其实Linux应该被称为 **GNU/Linux** 才对，我们去搜索 **GNU/Linux** 时，会发现 **一个牛头和一只企鹅** ，如下图：



看起来或许会很诡异，所以我们得先了解它们的历史。

Linux的诞生

回到我的C语言教程，其实最早的计算机是没有操作系统这个概念的，之后一个大佬为了玩游戏而移植了一个大型计算机的操作系统，用于运行他的游戏，而最后慢慢演化，才有了操作系统这种概念，最终铸就了 **UNIX** 的传奇，而最早 **Unix** 其实是免费的，开源的，每个厂商都能基于 **Unix** 配合自己的公司生产的硬件来开发一个专属于自己电脑的系统，既然是这样，还有 **Linux** 什么事呢？难道说 **Linux** 只是 **Unix** 的儿子吗？简单继承吗？不不，并非如此。

> Unix的前身为1964年开始的Multics，1965年时，贝尔实验室加入一项由通用电气和麻省理工学院合作的计划；该计划要建立一套多用户、多任务、多层次（multi－user、multi－processor、multi－level）的MULTICS操作系统。贝尔实验室参与了这个操作系统的研发，但因为开发速度太慢，1969年贝尔实验室决定退出这个计划。贝尔实验室的工程师，肯·汤普逊和丹尼斯·里奇，在此时自行开发了Unix。

美国AT&T公司的贝尔实验室 —— 研发出Unix的实验室

Unix的普及和声誉，使得AT&T公司嗅到了商机，于是Unix从公益项目转而变成了一个商业项目

> 此后的10年，Unix在学术机构和大型企业中得到了广泛的应用，当时的UNIX拥有者AT&T公司以低廉甚至免费的许可将Unix源码授权给学术机构做研究或教学之用，许多机构在此源码基础上加以扩展和改进，形成了所谓的“Unix变种”，这些变种反过来也促进了Unix的发展，其中最著名的变种之一是由加州大学柏克莱分校开发的伯克利软件套件(BSD)产品。
>
> 后来AT&T意识到了Unix的商业价值，不再将Unix源码授权给学术机构，并对之前的Unix及其变种声明了著作权权利。

这个时候，追求自由，公平的黑客就非常不满了，于是就开始了自己的小动作。当然，感到不满的，可不仅仅只有黑客们，还有教师，计算机狠人等等。于是，Linux系统和GNU计划就在这样的背景里出世了。

其实Unix系统有很多变种，其中最出名的就是 **BSD**、**Linux**、**Mac OS**（Windows不在Unix变种内，它是独立开发的，前身是MS-DOS系统），下图是一张调侃的梗图：

相信你已经对Linux由一定理解啦，就是Linux非常屌，Mac异常屌，Windows属于适用于大部分人的屌

我个人的长期体验后对于上面系统的简单评价就是：

1. WIndows：拥有无与伦比的兼容性，微软这b做向下兼容一直是可以的，但是动画有点生硬，从Win11开始似乎向着Apple靠拢，从圆角到棱角分明，再到Win11的圆角设计。
2. MacOS：拥有无与伦比的丝滑性，太优雅了，而且很适合用于开发，真的很人性化很优美很优雅，而且软件生态也很好，集成了Windows环境的软件生态和Linux命令的强悍。
3. Linux：事实上Linux有很多发行版，就连安卓都是基于Linux的再开发，Linux社区太强悍了，但是Linux事实上很多动画之类的处理不甚理想，而发行版的不同也导致了风格各异的Linux，所以Linux的体验感也因人而异，因发行版而异，但不得不承认，Linux是伟大的，默默奉献的。

Linux与盈利项目无关，它和WIndows闭源盈利不同，与Mac的盈利不同，Linux是有无数位无私的开发者，倾情维护的，开放源代码的（除了那个CentOS - yum系，前段时间停止维护，被大家骂背叛开源精神的盈利 RedHat-红帽公司，等）

GNU/Linux

讲也讲完了，骂也骂完了，那么是时候来说说，究竟什么是GNU项目了

> GNU操作系统起源于GNU计划，由理查德·斯托曼在麻省理工学院人工智能实验室发起，希望发展出一套完整的开放源代码操作系统来取代Unix，计划中的操作系统，名为GNU。1983年9月27日，理查德·斯托曼在net.unix-wizards 和 net.usoft新闻组中公布这项计划。理查德·斯托曼通过使用变化单词的各种手法来选择名称，包括曲目*The Gnu*。
>
> 斯托曼的目标是成立一个完全自由的操作系统，他希望电脑用户是能够"自由使用"的。因为在20世纪60年代和70年代 - 大多数人都能自由学习软件的源代码，自由地与他人分享的软件，可自由修改软件的行为，自由发布的软件的修改后的版本。这种理念，在1985年3月出版的GNU宣言崭露无遗。—— 最后他和律师共同起草了一份开源协议，这个协议就是之后大名鼎鼎的**GPL**协议了。

GPL协议：

> **GNU通用公共许可协议**（英语：GNU General Public License，缩写GNU GPL 或 GPL），是被广泛使用的自由软件许可证，给予了终端用户运行、学习、共享和修改软件的自由。

历史上，GPL许可证系列一直是自由和开源软件领域最受欢迎的软件许可之一。

而为什么Linux前面要加个GNU呢?

因为有软件而无系统始终是GNU项目的痛，而Linux反对Unix闭源，于是大量使用开源软件，那这些开源软件从什么地方来呢？就从GNU项目。但是GNU项目没系统，开发的软件又在什么地方用呢？是的，在Uinx上用，是不是很戏剧性，开源项目运行在商业项目上。所以这一直是GNU的痛，而Linux开发出来为了使用GNU项目软件，也是把自己兼容了Unix，使用上了GNU项目的软件。

所以其实Linux和GNU是相辅相成的，就有了 **GNU/Linux** 这种命名提案。

Linux基础

Linux命令概要 与 包管理器

哥们跳过理论，直接上基础，因为理论其实很乏味的，这里只作教程，让你快速上手Linux，而不考虑后话。

但是还是有必要了解一下，何为APT系，何为YUM系。

其实APT系，就是指比如以Debian，Ubuntu之类为首的Linux发行版，此类发行版采用的包管理器是APT，所以安装软件的时候会使用到这样的命令：

```
apt install xxxxx
```

那YUM系又是什么呢？就是以CentOS，Fedora之类为首的Linux发行版，此类发行版一般用yum命令来安装软件，也就是采用了YUM软件包管理器的系统：

```
yum install xxxxx
```

其实还有一些其他的包管理器，比如大名鼎鼎的Pacman(Arch)，Portage(Gentoo)，ZYpp(openSUSE/SUSE Linux Enterprise)

甚至MacOS下的Homebrew包管理器，这些包管理器提供了非常快捷的安装使用软件方式，当你还在用繁琐的下载源代码，编译安装时，我早已通过 `apt install` 代码安装上我所需要的软件。

源 - 包管理器身后的老大哥

我们是否可以把apt/yum之类的包管理器就理解成一个应用商店呢? 不严谨地说，是可以的。

我们这里引入一个「源」的概念，发行版的包管理器，有一个文件用于管理一些小网址，这些小网址可不是给你这byd涩涩用的😡，而是 ——「源」，**究竟什么是源呢？源就像一条小路，而应用商店就是一片汪洋，这一条条记录在本地计算机的记录，写在一个文件里面，这个文件就是一张地图，指引我们的Linux去到这片汪洋**。

就像我们手机的应用商店没有网络无法打开一样，其实它上面的内容也都是向服务器请求下来的，服务器记录哪个App最受欢迎，App分类，当我们连接网络，打开应用商店，就能看到服务器上面同步下来到我们手机的内容，此时我们点击下载，就能快速下载并安装，而不需要上到Baidu去找，甚至会有可能找到一堆病毒或者打不开的（损坏）软件回来。*（如果没有网络也能显示，那就是因为你系统内置缓存了，或者你前段时间用过了，缓存下来的内容，下次打开即便没有网络也能看到里面的内容，但是你却不能下载，因为没有网络，由于是之前的数据，“今天最受欢迎的App”之类的功能也会变得不准确，因为这些数据已经是最后一次更新的数据了，谁知道你最后一次打开，它获取数据并缓存下来是半年前还是几天前，还是今天早上呢？）*

所以源其实有一个命令，所有新系统安装完毕都应该执行一下，因为你的镜像可能是今年5月的，而现在已经是9月了。~~那么长时间，可能原神都从2.0版本升级成4.1版本了。~~ 所以，下面**这两条命令就显得极为重要**了：

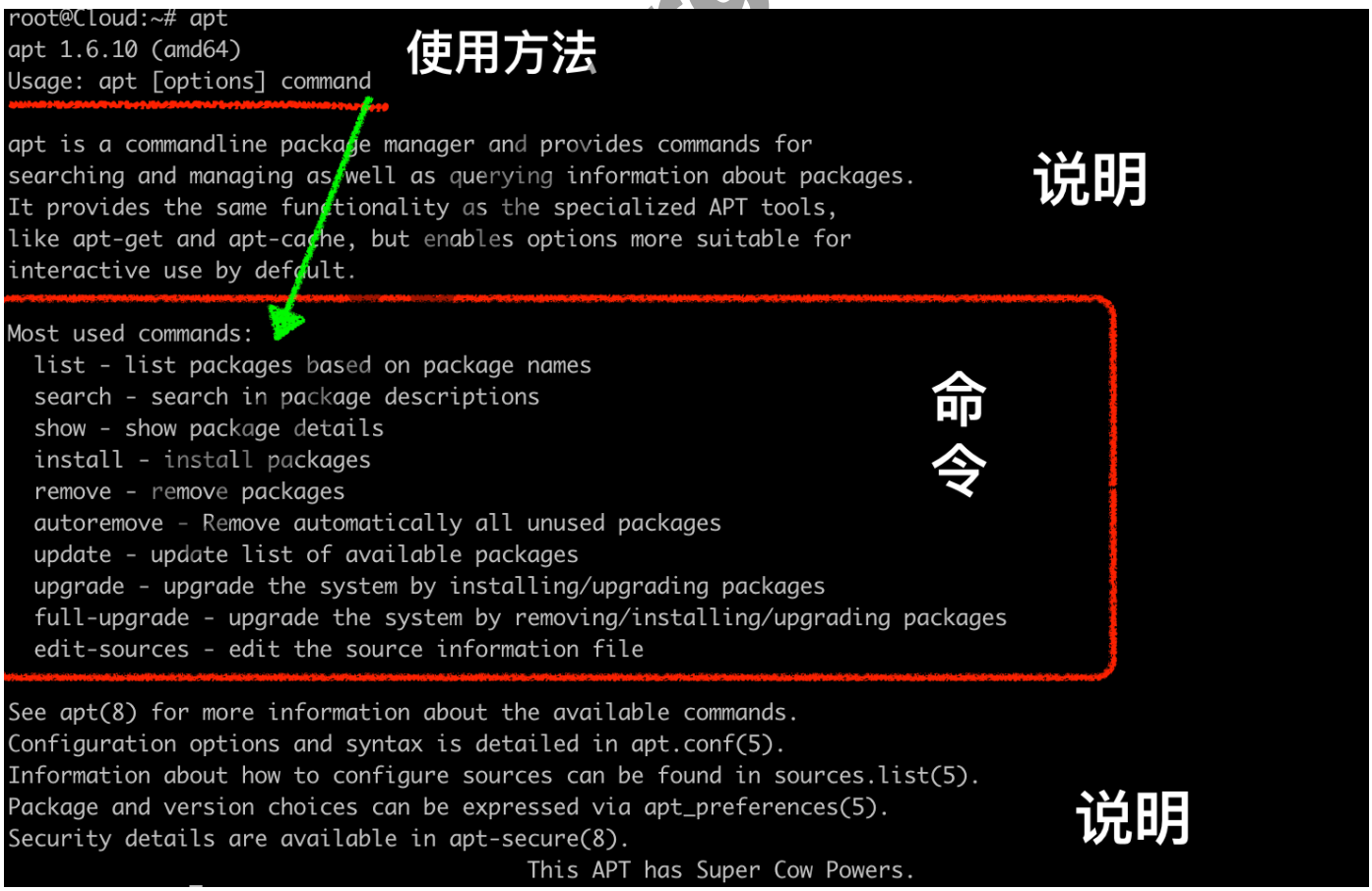
```
apt update
apt upgrade
```

> - `apt update`：该命令用于更新本地软件包索引。它会从配置的软件源中下载最新的软件包列表信息，以便了解可用的软件包及其版本。简而言之，`apt update` 只是更新本地软件包信息，并不实际更新软件包。
> - `apt upgrade`：该命令用于升级系统中已安装的软件包。执行 `apt upgrade` 命令将会根据本地软件包索引中的信息，检查已安装软件包的可用更新版本，并对其进行升级。这会将已安装的软件包更新到最新版本，包括安全更新、修复漏洞和提供的新功能。
>
> 所以，`apt update` 用于更新软件包索引（即更新可用软件包的列表），而 `apt upgrade` 用于升级已安装软件包到最新版本。一般来说，先执行 `apt update` 更新软件包索引，然后再执行 `apt upgrade` 升级系统中的软件包。这样可以确保系统获得最新的软件包信息并进行升级以保持系统安全和功能最新。

当然yum和Pacman等包管理器也都一样，几乎都是大同小异，特别是yum。

其实如果我们想要知道一个代码在Linux下有什么参数/方法使用的话，我们可以输入 `[命令] -h` 或者是 `[命令] -help` 或者是 `[命令]`

假如我们输入`apt`它会返回



这里就已经把`apt`命令的使用说明和参数列得很明白了。这里的「 apt 」指定了我们要使用「 apt 」这个程序，而`apt install [包名 或者说 程序名]`里面的「 install 」就是图上的命令，它已经写得很明白啦 - 「install - 安装 一个包」，所以我们使用`apt`去安装，比如说Nginx这个程序的时候，就得使用这行代码`apt install nginx`，如果是`yum`则是`yum install nginx`。上面还写明了，可以使用`remove`去移除（卸载）一个包（程序），可以使用`search`命令去搜索某个程序是否在「 源 」里面，可以使用....等等。

```
apt install nginx     ##Apt系的Linux发行版 —— apt:使用apt这个程序  install:安装  nginx:名为nginx的包
```

或许跟着操作的你，细心的话，就已经发现有点不同了：

为什么我的终端（Bash）和你的并不一样？你的「 root@Cloud~# 」处，和我的区别那么大？

不仅仅字不一样，而且我这个黑到难以看得出来？

因为我们用的发行版不同，终端程序不同，甚至“身份”也不同

身份 - Root（管理员）

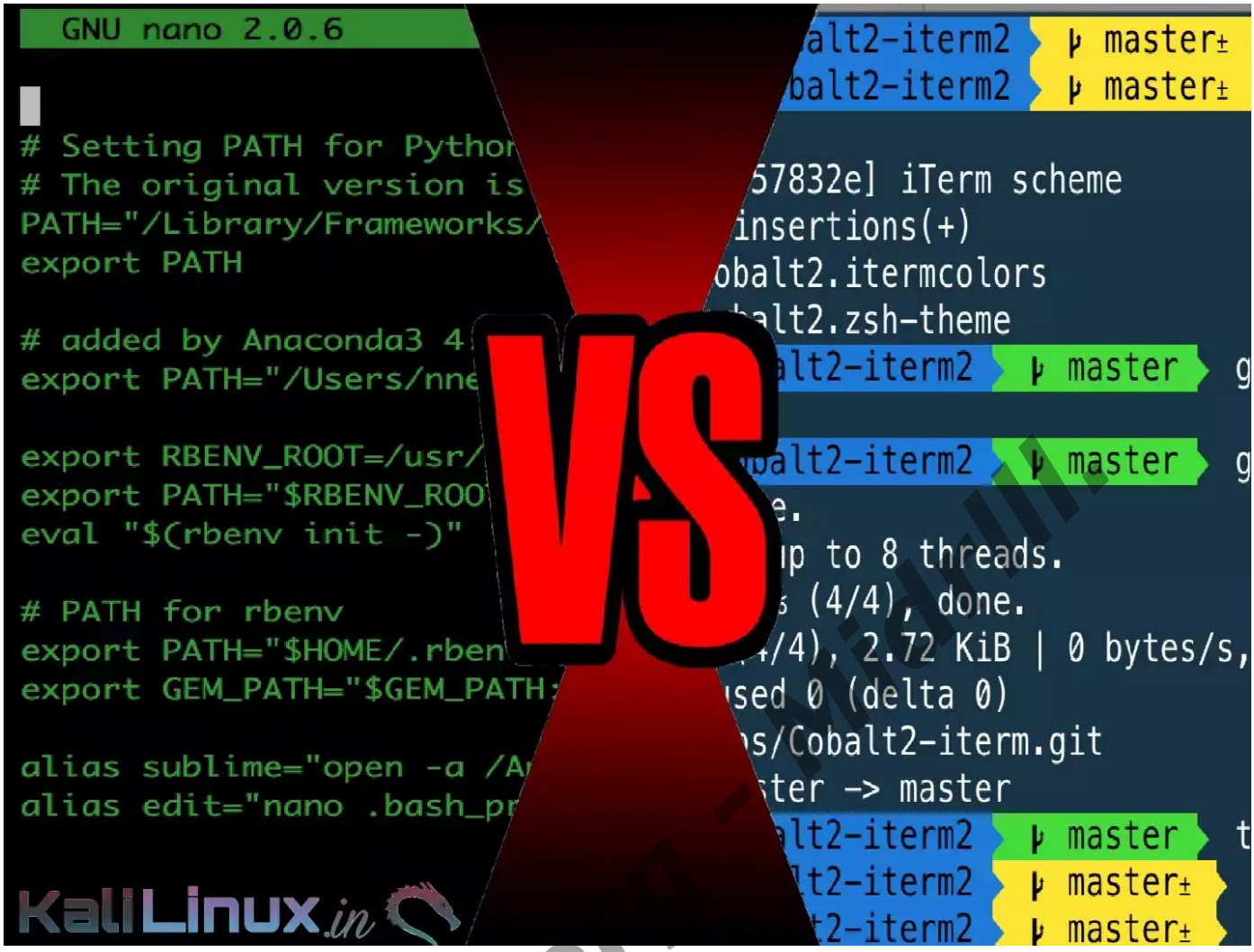
在一般情况下，终端的显示是这样的`root@Cloud~$`

在管理员身份下，终端的现实却是这样的`root@Cloud~#`

「 # 」和「 $ 」为了让你知道，自己究竟是什么身份，普通用户还是管理员

而使用zsh会更好看，bash会相对单调些，而我因为是在bash终端内使用ssh命令连接了服务器，所以看起来更简陋，甚至连高亮显示都没有了。

下图左边是bash，右边是zsh：



很显然，bash更酷（原始的酷），而zsh非常优美（zsh有很多主题）

但是当我们说完好看与否这个问题时，我们不妨回到，到底什么是管理员？它和普通用户有何区别？

区别很大，其实在你输入`su`（使用管理员身份）时，它就会告诉你：

```
root@Cloud~$ su
We trust that you have already learned the daily dos and don'ts from your system
administrator.
To summarize, it's nothing more than these three points:

    #1) Respect the privacy of others.
    #2) Think before you type (consequences and risks).
    #3) With great power comes great responsibility.

[su] xxx Passwords:

########################## 中文显示 ##########################

root@Cloud~$ su
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。

[su] xxx 的密码:
```

此时，它会向你索取你的登录密码（root登陆密码），但是请注意 **Linux涉及到密码的输入是不显示的，就是说，即便你输入了，它也不会提示任何内容，但是有些发行版会提示「****」，用来告诉你，你有在输入密码**。

当你密码输入正确，回车时，`$`标识变成`#`时，就证明你已经获得Root管理员权限了，此时你就可以随意访问「/」系统根目录，甚至强制删除整个系统了：

```
sudo rm -rf /*    ## 这应该是个笑话，而不是让你自己成为笑话。这行代码的解释是：[sudo]使用管理员权限，[rm]删除命令，[-rf]删除命令的强制删除参数，[/*]指定了rm要删除的目录，/ 代表根目录，* 代表所有。| 使用管理员权限，强制删除，根目录下所有文件。
```

常用的Linux命令

让我们看看下面这些常用的Linux命令: 非常重要的已经加粗显示

1. 文件和目录操作命令：
    - **`ls`: 列出目录内容**
    - **`cd`: 切换目录**
    - **`pwd`: 显示当前目录的路径**
    - **`mkdir`: 创建新目录**
    - **`rm`: 删除文件或目录**
    - **`cp`: 复制文件或目录**
    - **`mv`: 移动文件或目录**
    - **`cat`: 查看文件内容**
    - **`touch`: 创建空文件或更新文件时间戳**
    - **`find`: 在文件系统中查找文件或目录**
2. 系统管理命令：
    - `top`: 显示系统资源使用情况
    - **`ps`: 显示运行的进程信息**
    - **`kill`: 终止运行的进程**
    - `df`: 显示磁盘空间使用情况
    - `du`: 估算文件和目录的磁盘空间使用量
    - `free`: 显示系统内存使用情况

- `ifconfig`: 显示和配置网络接口信息
- **`ping`: 测试网络连接**
- **`wget`: 下载文件**
- `ssh`: 远程登录到其他计算机
- **`shutdown`: 关闭系统**
- **`reboot`: 重启系统**

3. 压缩和解压命令：
   - **`tar`: 打包和解压文件**
   - `gzip`: 压缩文件
   - `gunzip`: 解压缩文件
   - **`zip`: 压缩文件**
   - **`unzip`: 解压缩文件**
4. 权限管理命令：
   - **`chmod`: 修改文件或目录的权限**
   - **`chown`: 修改文件或目录的所有者**
   - `chgrp`: 修改文件或目录的所属组
5. 文本处理命令：
   - **`grep`: 在文件中查找匹配的文本**
   - `sed`: 流编辑器，用于处理和转换文本
   - `awk`: 文本处理工具，用于提取和处理数据

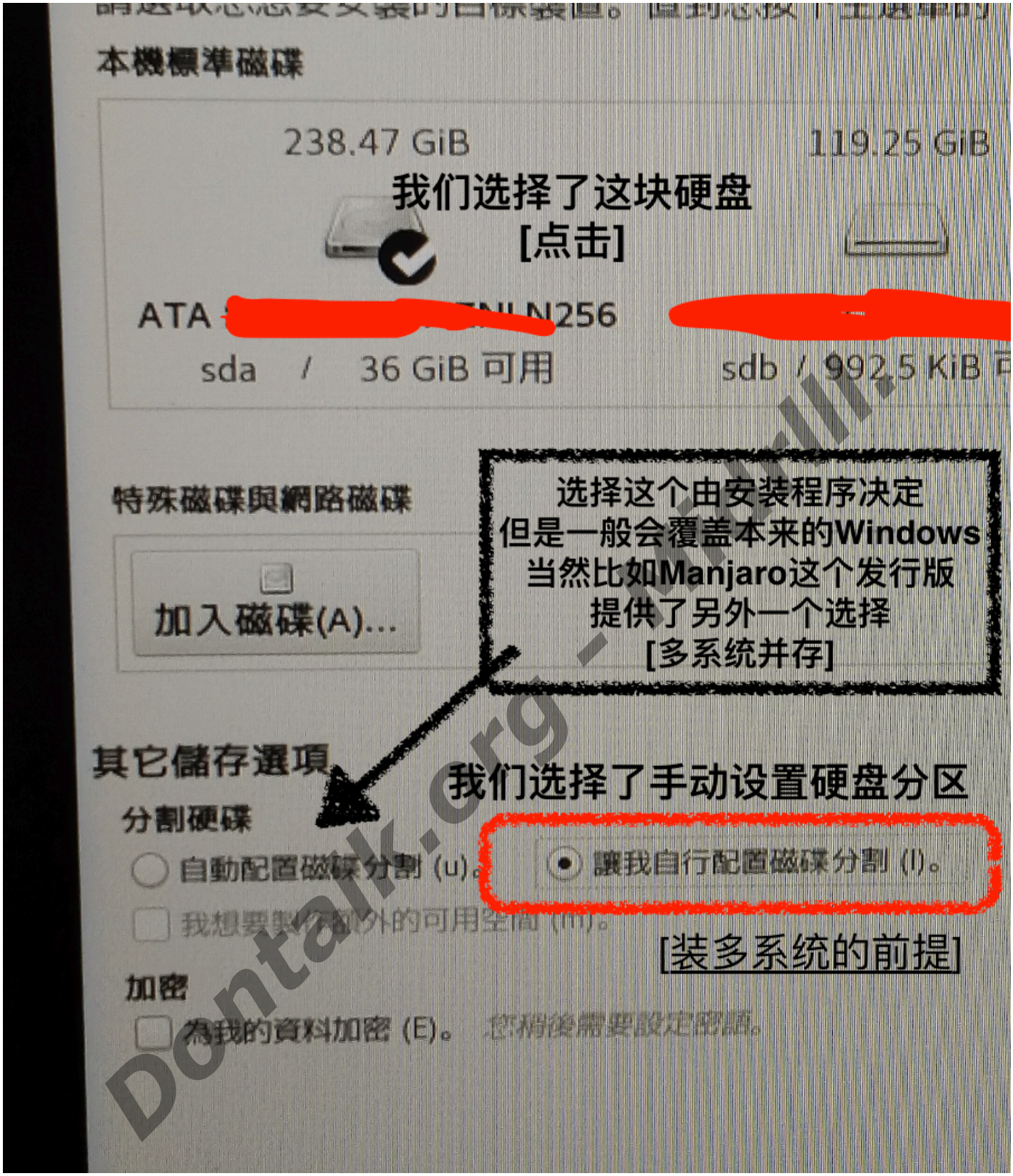
这只是一部分常用的 Linux 命令，Linux 系统提供了更多强大的命令和工具，可以根据需要扩展和深入学习。

Linux安装值得注意 - 硬盘分区

在我初中第一次安装Linux，框框一把梭，但是在硬盘处却被难住了，最终安装失败。。

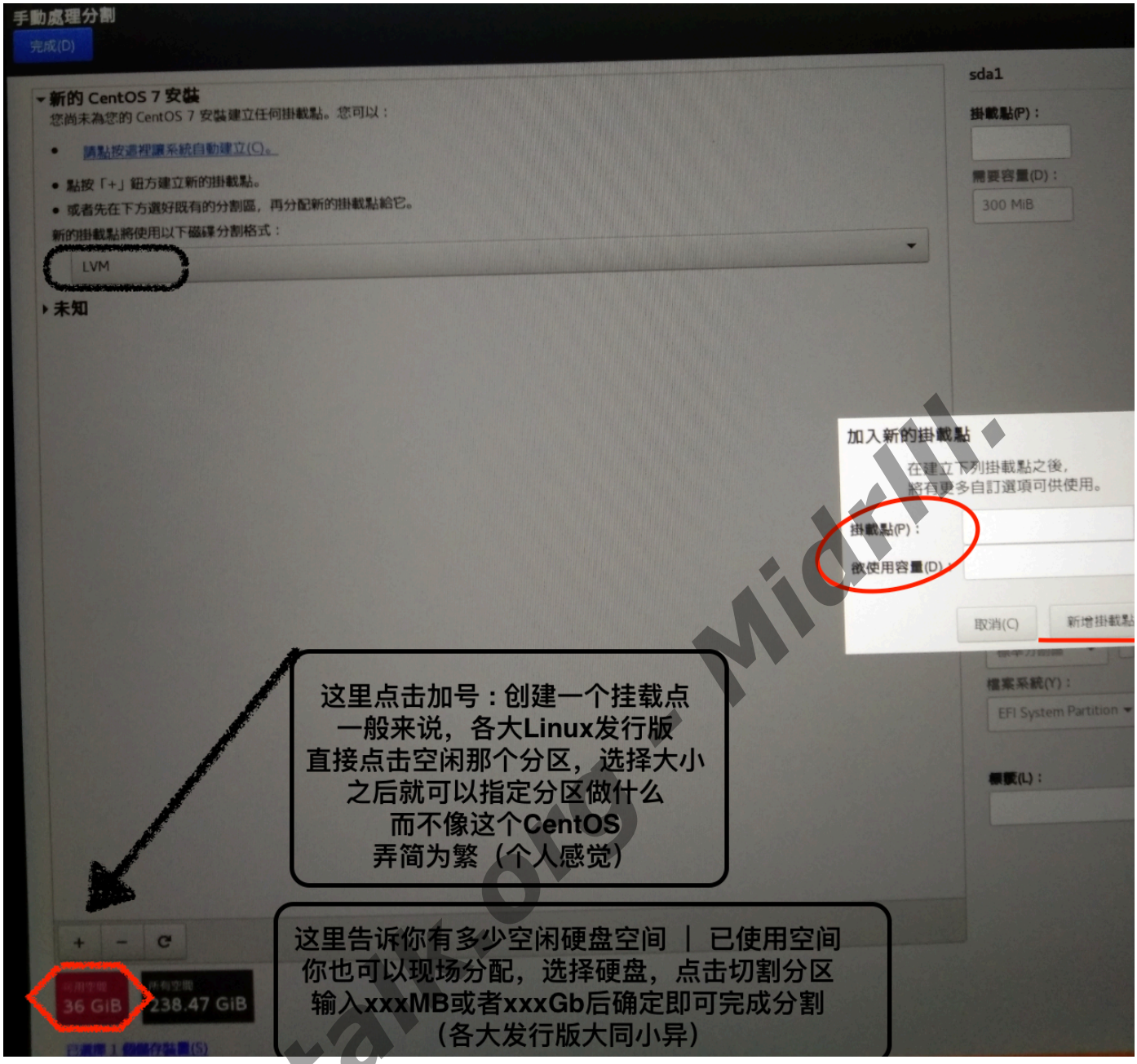
可以说，Linux安装里，最困难的或许就是硬盘分区设置了（Arch，Gentoo安装除外）

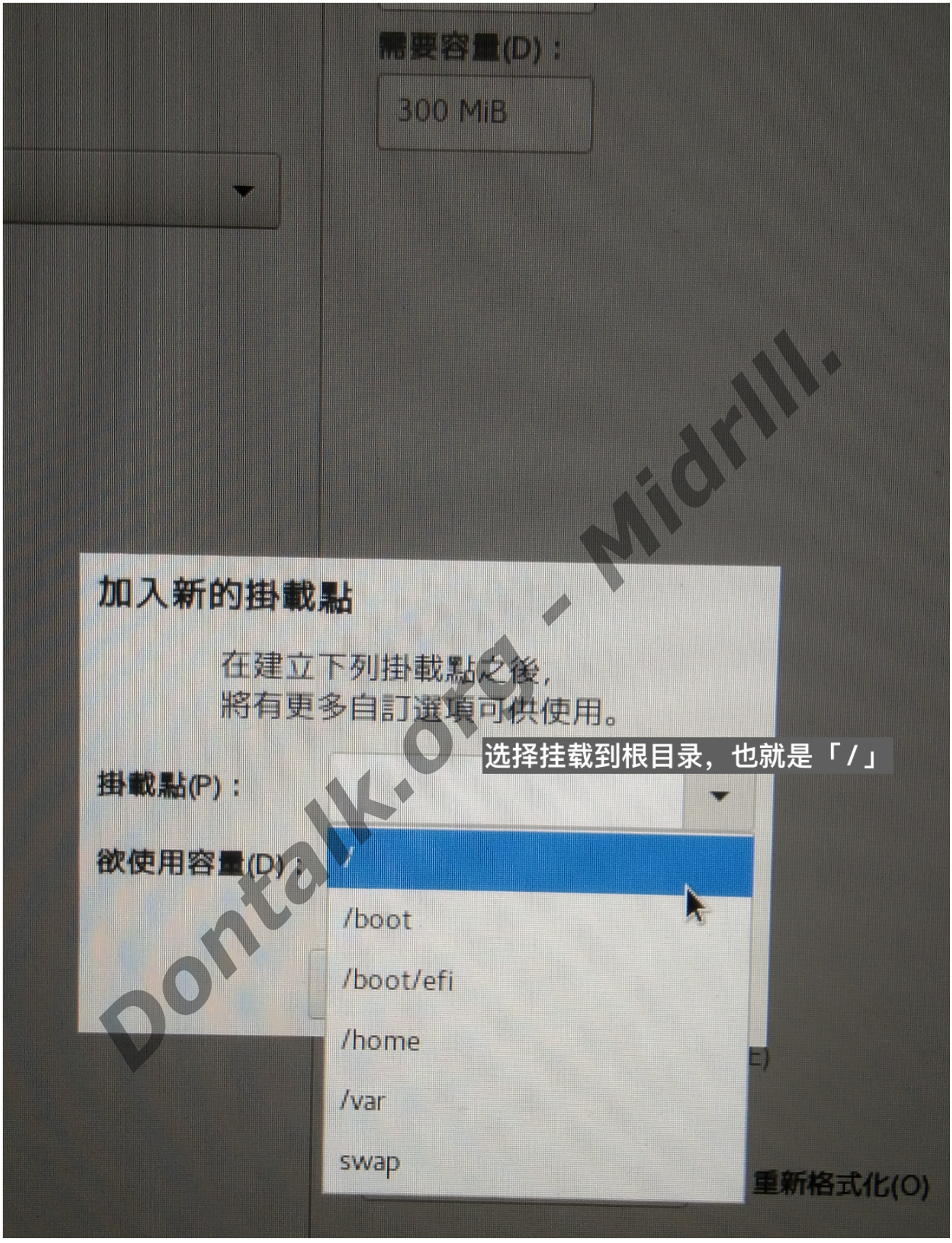
我们直接快进到安装硬盘这一页：（*CentOS作演示，其余发行版其实都大同小异，你要做的就是理解*）

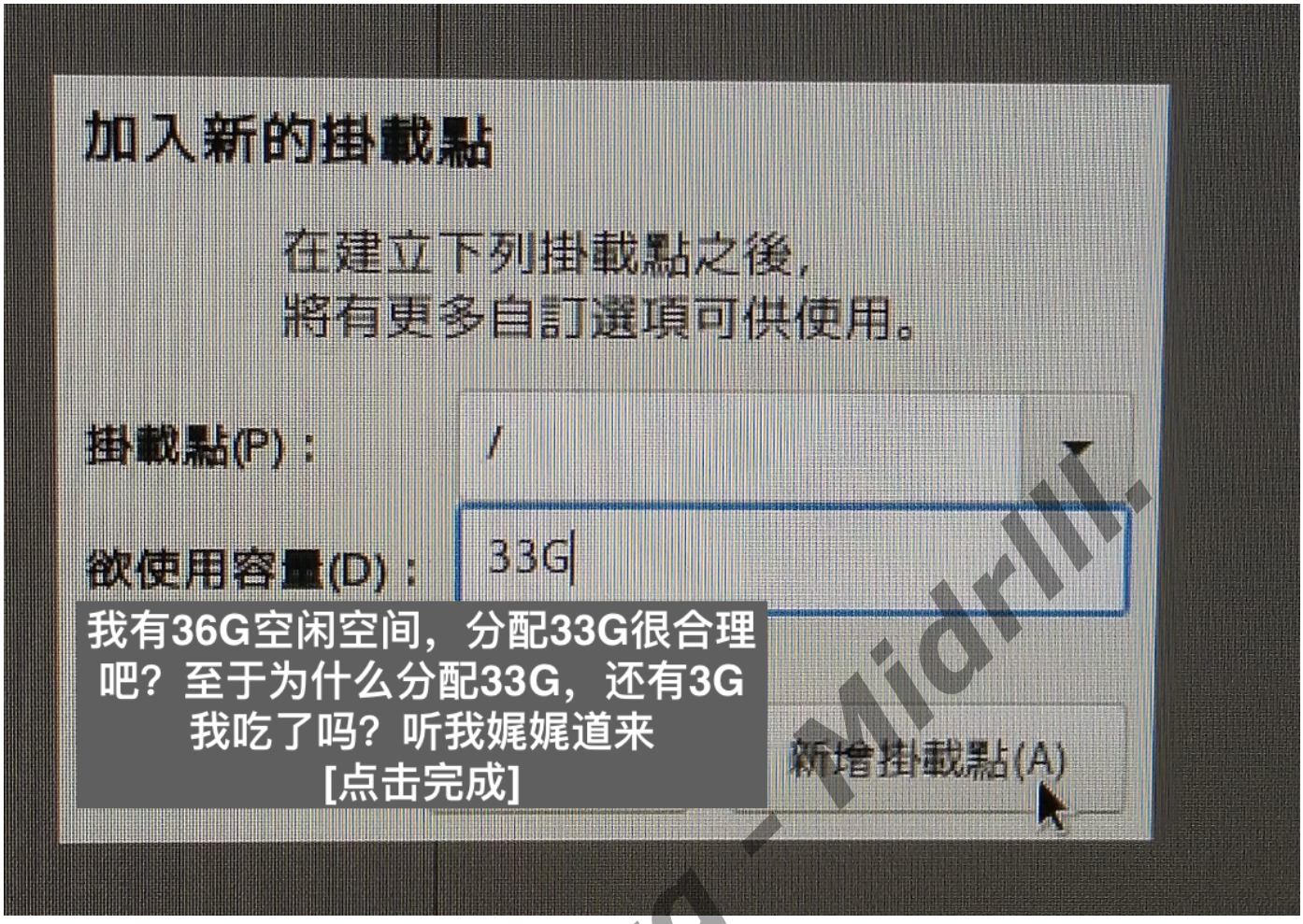


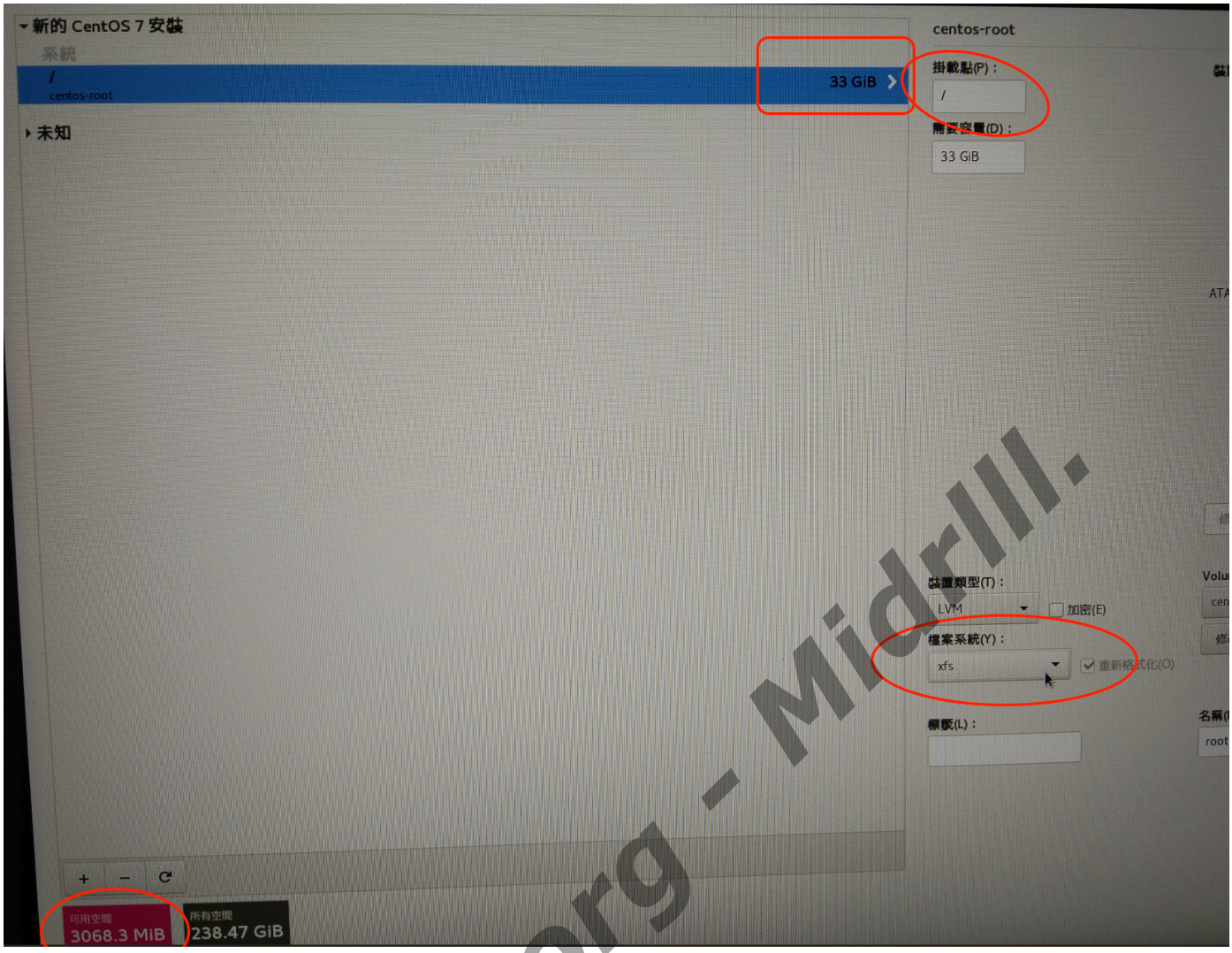
加密硬盘这个选项其实一般人用不上，OK的，让我们点击下一步：

挂载系统根目录（挂载系统本身）

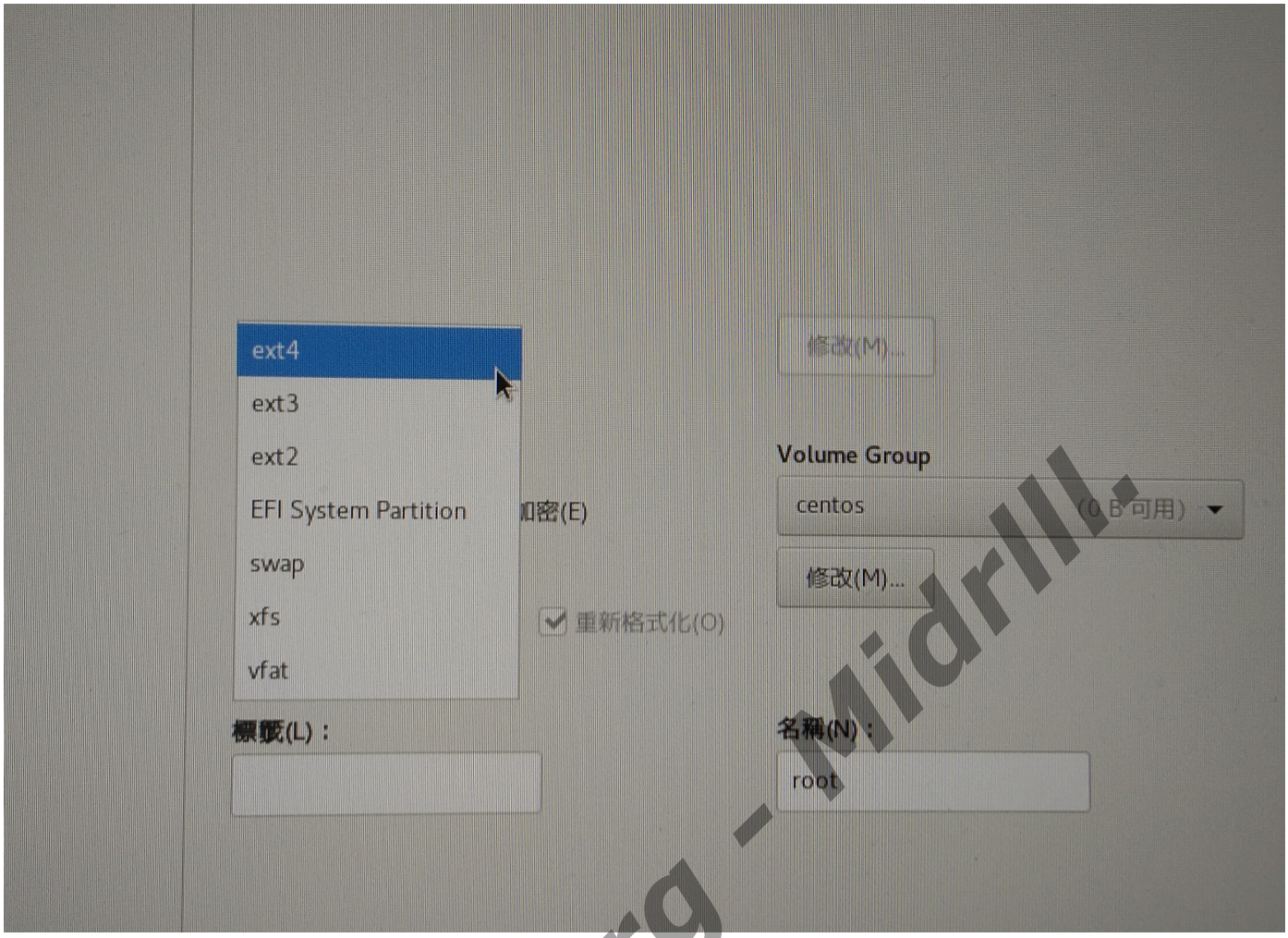








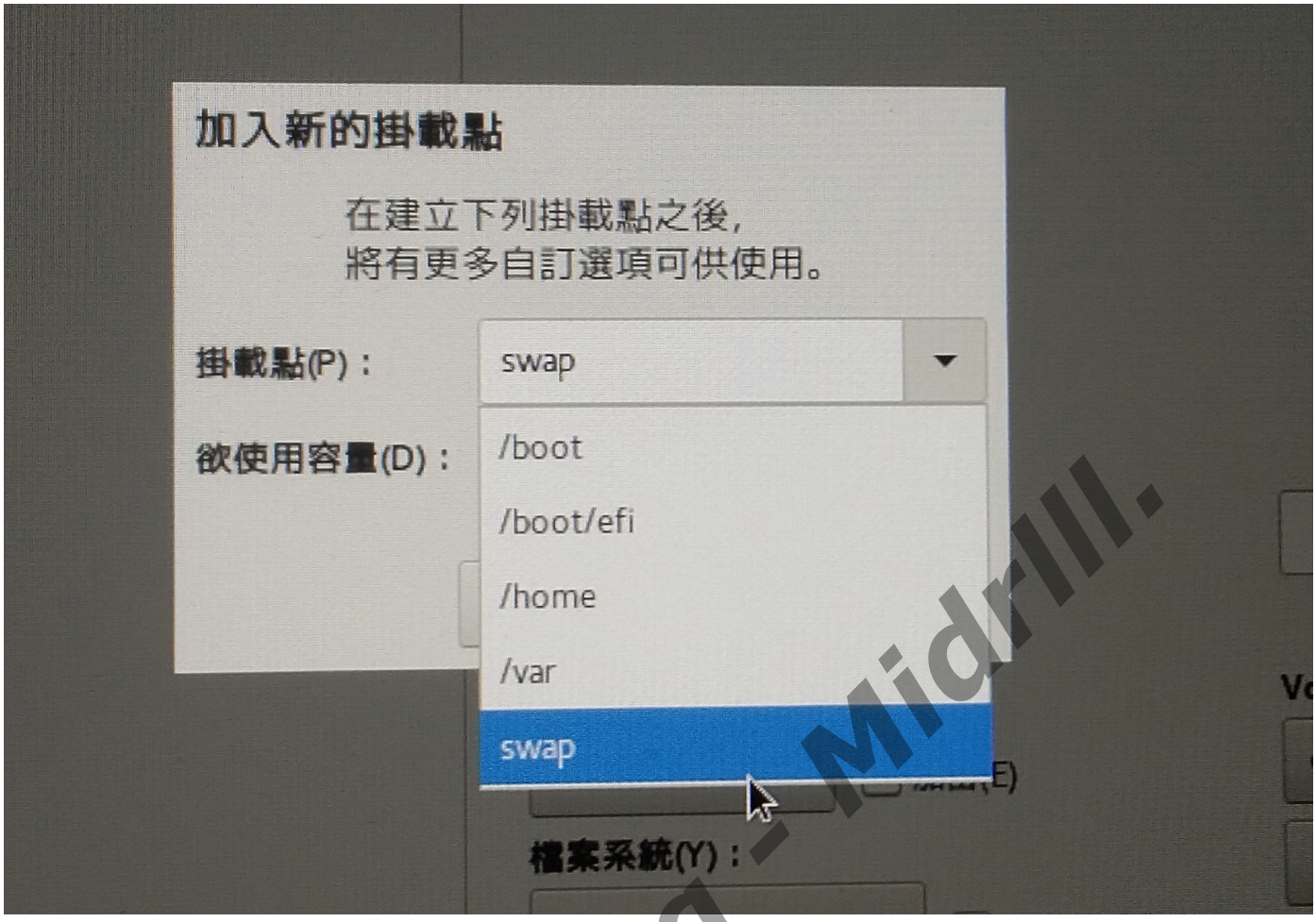
此时我们还需要更改右下角圆圈，档案系统，更改成「 EXT4 」，记得挂载位置在 根「 / 」：

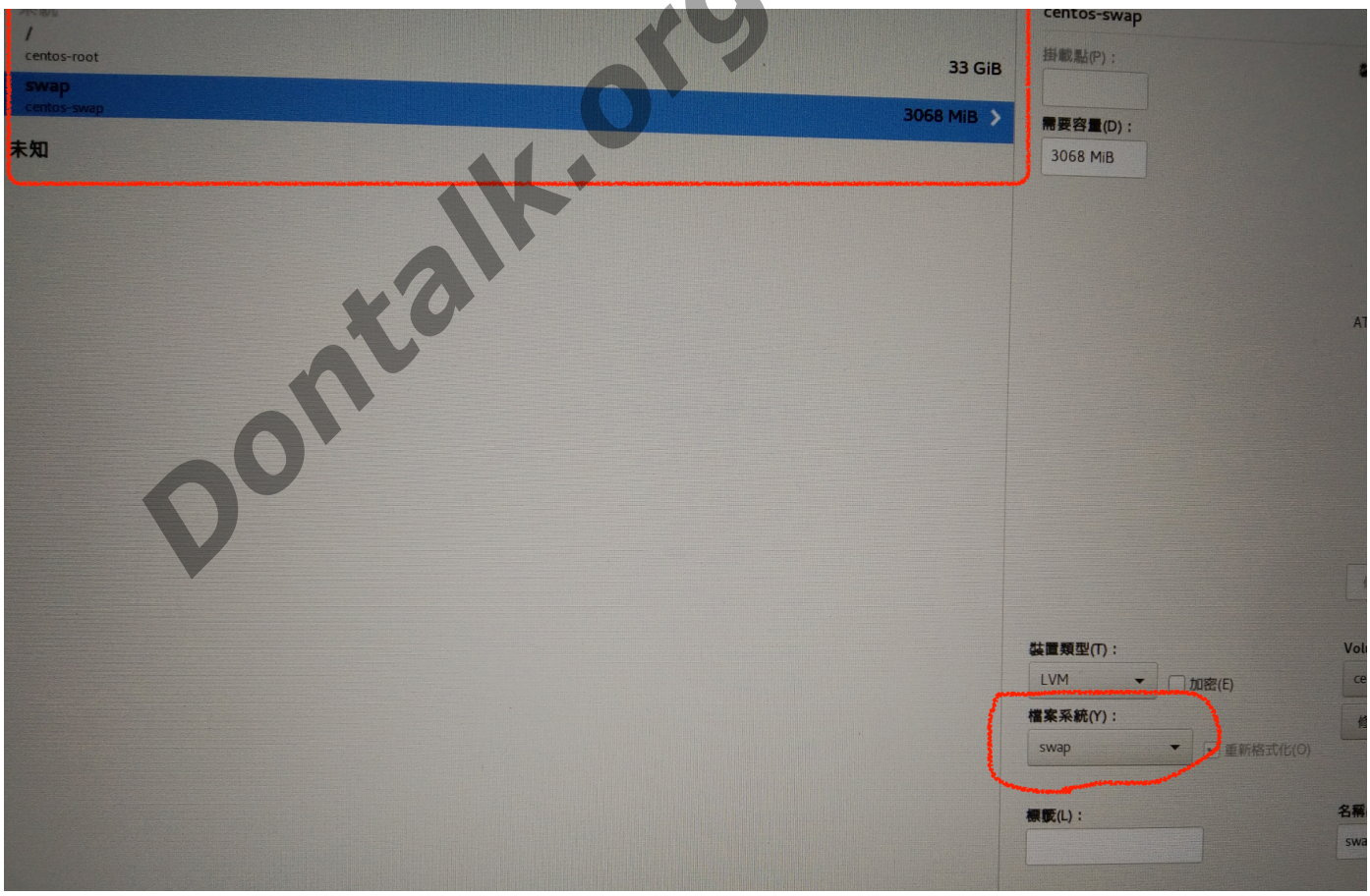


完成所有后，我们继续在剩下的3G磁盘空间中创建分区

挂载系统交换分区（用硬盘大小充当运行内存用）

此时我们可以看到空闲空间剩下3G了，我们继续上面的操作，但是分区挂载点得换换：（换成swap，SWAP就是Linux交换分区的意思）





这样即可点击完成分区 - 确定完成并写入分区

这样就完成我们的Linux硬盘分配问题啦，但是这个交换分区，也就是硬盘做运行内存还是很有必要说一下，因为**交换分区并不是越大越好**，12G运行内存的手机也只是给了6-8G的交换空间，一般4G运行内存，给2G这样就足够了，太大其实用不上，也没必要。所以我即便电脑运行内存是32G，我也是只给了3G的交换空间，其实为了好看，你可以给4G，6G..这样8G的话就有点浪费了。🤔

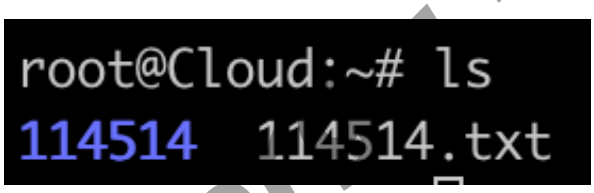
因为Linux大部分应用都比较安分，32G的运行内存其实家用完完全全就够用了，你还要多给那么多，不仅仅占硬盘空间，还没多大作用，所以实在犯不上，我服务器 1核心CPU，512Mb内存，218Mb交换空间，17Gb硬盘依旧玩得不亦乐乎，其实大部分工作都是CPU在做，做很多事情都在跑CPU。CPU好比内存大更吃香（看个人需求），总不能让1-2核CPU带32G内存，这跟吕布骑狗有何区别？🐶

Linux正式入门

此时，我们对Linux已经有了一定的了解，那么我们就开始试试将平常的工作交予Linux - Bash终端去做吧。为我们创建文件夹，输出文件，编辑文档。

列出该目录下所有文件：`ls`

当我们处于一个目录下，其实并不知道这个目录下究竟都有什么，此时我们就需要`ls`命令来列出该目录下的所有文件，就像我们打开Windows某个文件夹，你一眼就能看到里面有什么文件，但是Linux看不到，你使用了`cd [文件夹名称]`进入文件夹后，还得通过命令`ls`列出该目录下所有文件，才能知道该目录下究竟有什么内容。



我们此时可以看到`:~/`（~为用户目录，一般存放“桌面”、“文档”、“下载”等文件夹）下存在两个文件，其中一个是名为“114514”的文件夹，一个是名为“114514”的txt文本文件。

如果我们想查看更多信息，可以为`ls`命令加上一些参数：`ls -la`*(列出文件详细信息，包括隐藏文件)*，下图中「.」开头即为隐藏文件，平常是看不到的，在Windows下也得开启显示隐藏文件才能看到。

```
root@Cloud:~# ls -ls
total 4
4 drwxr-xr-x 2 root root 4096 Dec 14 00:15 114514
0 -rw-r--r-- 1 root root    0 Dec 14 00:16 114514.txt
root@Cloud:~# ls -la
total 60
drwx------  6 root root 4096 Dec 14 00:16 .
drwxr-xr-x 23 root root 4096 May  3  2019 ..
drwxr-xr-x  2 root root 4096 Dec 14 00:15 114514
-rw-r--r--  1 root root    0 Dec 14 00:16 114514.txt
-rw-------  1 root root  703 Dec 14 01:06 .bash_history
-rw-r--r--  1 root root 3106 Apr  9  2018 .bashrc
drwx------  3 root root 4096 Nov 26 19:23 .cache
drwx------  3 root root 4096 May  3  2019 .gnupg
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
-rw-------  1 root root    7 Nov 26 19:21 .python_history
drwxr-xr-x  2 root root 4096 Nov 26 14:18 .ssh
-rw-------  1 root root 7419 May  3  2019 .viminfo
-rw-r--r--  1 root root   17 May  3  2019 .vimrc
-rw-r--r--  1 root root  215 Nov 26 14:26 .wget-hsts
-rw-------  1 root root   51 Dec  6 11:23 .Xauthority
root@Cloud:~#
```

事实上，很多代码都有参数这样说，比如说`apt`，其中`apt install`里的「install」即为指令参数（其实这里用命令来说更严谨一些，因为参数一般以`-x`的形式出现）

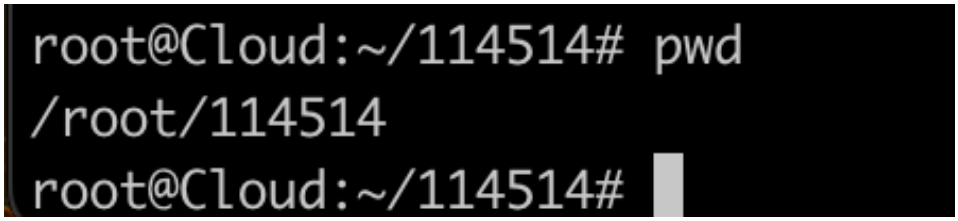
进入文件夹命令：`cd`

```
-rw-------  1 root root   51 Dec  6 11:23 .Xauth
root@Cloud:~# cd 114514/
root@Cloud:~/114514# ls
root@Cloud:~/114514#
```

我们可以看到，进入了114514这个目录后，我们的`:~#`就变成了`:~/114514#`，可以看出来我们已经身处114514这个目录内，而我们使用了`ls`命令列出该目录下所有文件，却没有发现任何文件，是的，这个文件夹下没有任何文件。

查看所在目录的地址：pwd

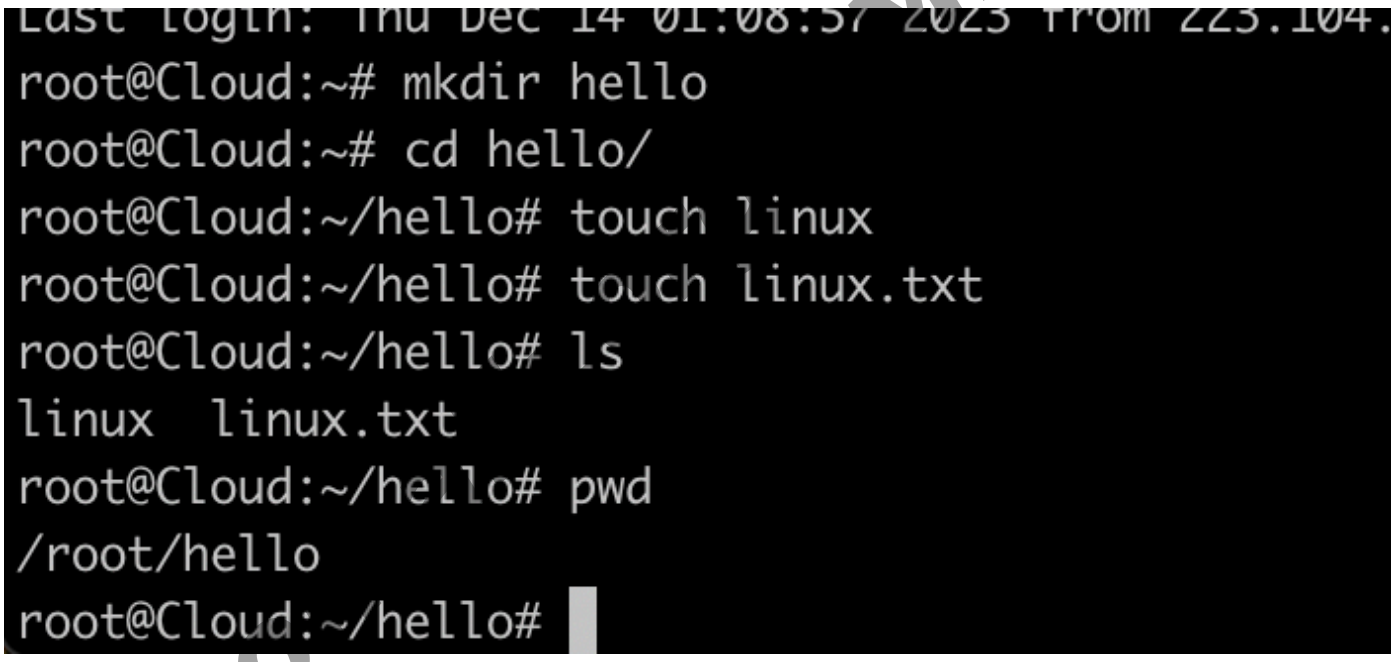
有时候我们不能知道自己到底在什么目录下，我们就可以用到该命令`pwd`



我们可以看到，我们正处于`/root/114514`目录下，我前面有说，「~」指的是用户的根目录，一般存着“桌面”、“文档”、“下载”等文件夹，是的。但是由于我是服务器，一直用着的用户就是「#(管理员)」，又正是因为是服务器，所以没有“桌面”、“文档”、“下载”等文件夹，所以就可以解释为什么我在`~`目录下使用`ls`命令为什么列出只有名为114514的文件夹和文本文件，也可以说明为什么我的`~`地址`pwd`显示的是「root」(`/root/114514`)。

创建文件、创建文件夹、删除文件(夹)：touch、mkdir、rm

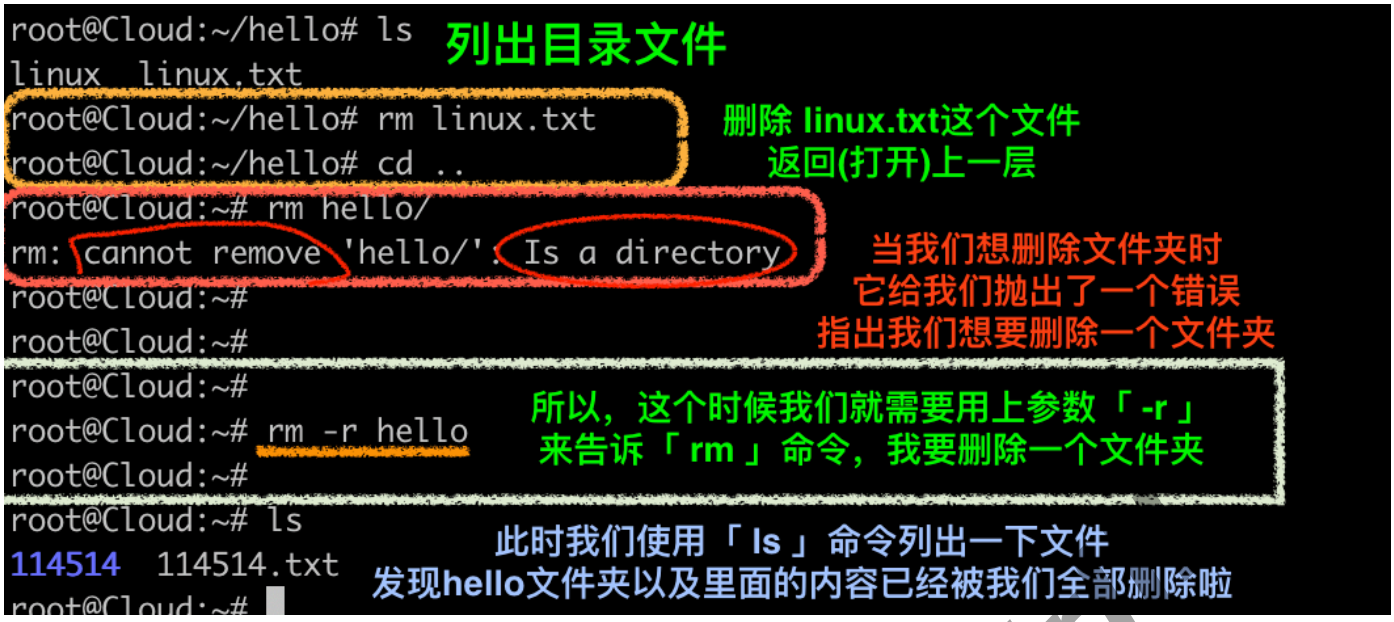
让我们来创建一个名为“hello”的文件夹，在“hello”文件夹里面创建一个名为“linux”无后缀文件，和一个有后缀的txt文本文件。之后再删除一个创建的文件，再回到`~`一次删除创建的文件夹和内所有文件。



我们通过`mkdir`创建了一个名为“hello”的文件夹，并分别使用了`touch`命令，产生了一个名为「linux」和一个名为「linux.txt」的文件，之后我们使用`ls`命令列出当前目录的所有文件，发现确实没错。我又使用了`pwd`命令，查看自己在哪一个目录。结果显示为`/root/hello`，一切正常，啊哈。

那么我们就要来操作rm命令了，**请记住，rm命令在sudo(管理员)情况下，权限无限大，甚至可以删除整个系统。所以在管理员权限下请谨慎使用，也要谨慎使用「-rf」(强制删除)参数**。

那么我们试着先删除「linux.txt」这个文件，再回到上一个目录，`~`(root)目录一次删除整个文件夹，包括里面的文件。



提示：如果我们想创建一个文件夹，在hello里面的123文件夹，而hello没有创建，我们是否可以这样去创建文件夹呢?

```
mkdir hello/123
```

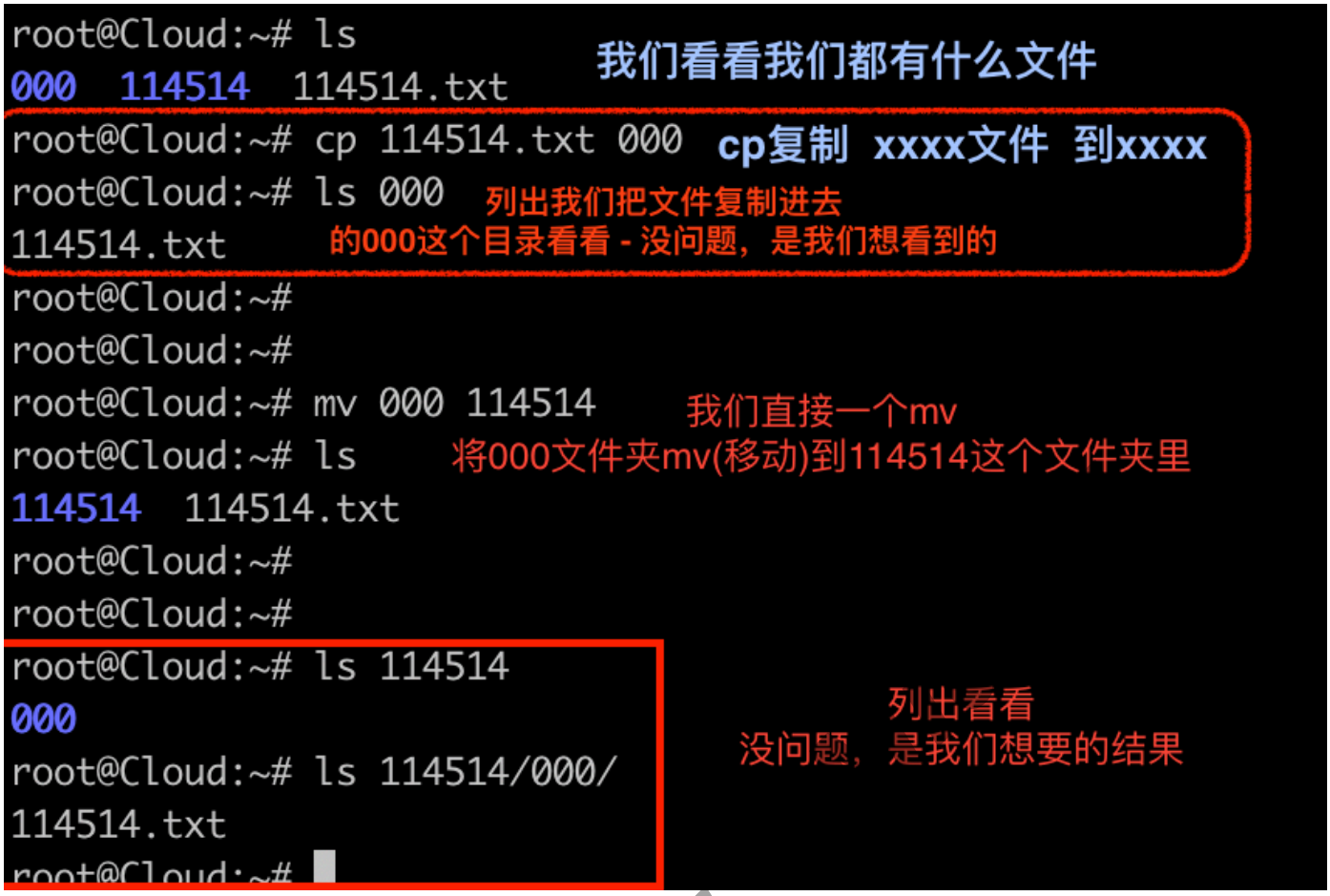
当然是可以的，但是不出所料的话，又将报错。为什么呢？因为你遗忘了Linux命令下很重要的参数这一设定，当我们把代码换成下面这样的话，就可以完美运行并创建出来了：

```
mkdir -p hello/123
```

复制和剪切移动：cp、mv

我们的「 root 」(~)目录下有三个文件，分别是114514.txt这个文件，和114514这个文件夹，和000这个文件夹。

我们要做的就是，将114514.txt这个文件复制到000这个文件夹，再将000这个文件夹移动到114514这个文件夹里：



寻找命令：find

万里寻它根目录，嫣然回首，结果他妈就在root用户目录处。😅

确实烦，不像Windows点进去能看到还好说，但是即便是方便如Windows，还是会需要用到搜索文件功能，那么我在Linux终端下该如何使用find搜索一个名为「114514.txt」的文件呢?

```
find / -name 114514.txt  ## 这段代码的意思是，使用find命令，在「 / 」根目录下搜索(相当于搜索整个系统)，-name 名字参数，搜索名为「 114514.txt 」的文件。
```

我们直接复制上面的代码到bash里面试试（## 后面为注释，bash并不会执行）

```
root@Cloud:~# find / -name 114514.txt ## 这段代码的意思是，使用find命令，在「 / 」根目录下搜索，-name 名字参数，搜索名为「 114514.txt 」的文件。
/root/114514/000/114514.txt
/root/114514.txt
```

我们可以看到搜索出来了两个，第一个在「114514/000」文件夹里面那个就是我们刚刚「cp(复制)」出来的产物了，于是`find`命令就给搜索出来了。

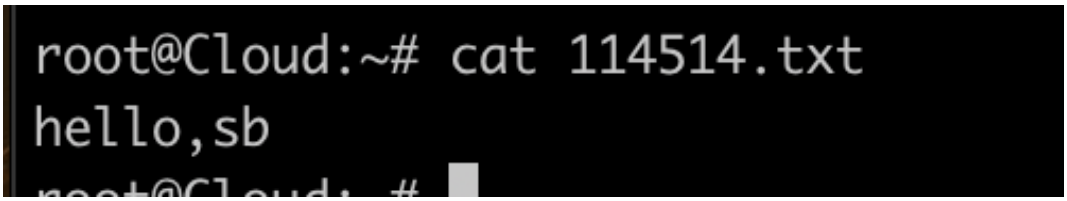
但是..但是呢，我们只知道我们要找的文件是「.txt」后缀，而不知道文件名称呢？怎么办啊？

拜托，你是学Linux的，来点拓展思维好不好，Linux的find命令当然可以解决啦!

```
find / -name "*.txt"
```

这段代码我就不解释了，自己运行和理解一下（此时，根目录所有能找到的「 .txt 」后缀文件都被输出出来）

查看文件到底写了什么：cat



我们可以看到「 114514.txt 」这个文件里面写了简短有力的一句话：“hello,sb”

话都说到这里了，我们该如何操作一个文件，修改里面的内容呢?

比如说C语言，py，txt等等

Vim太难了，为了入门，这里教学：nano文本文件编辑器

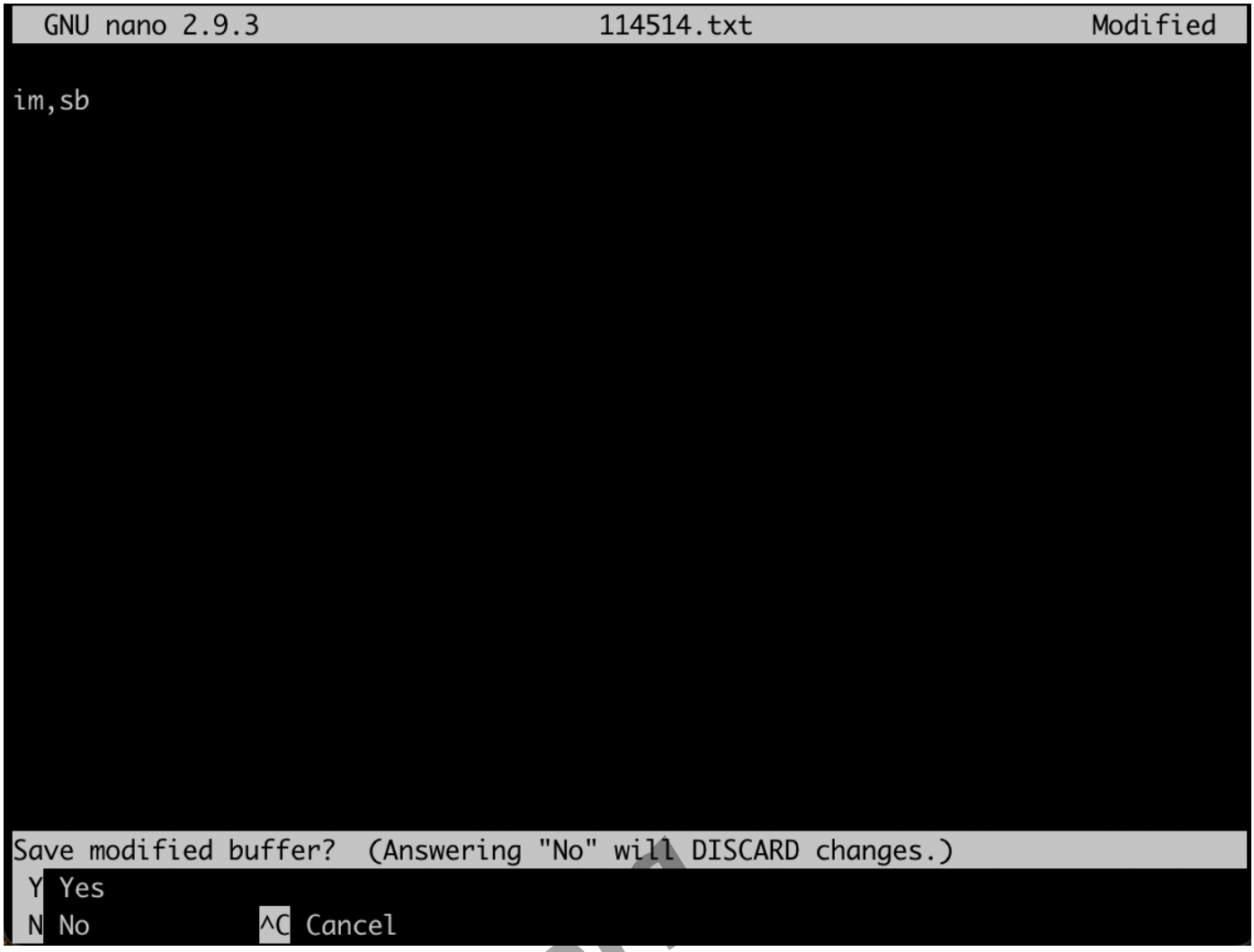
什么是文本文件呢？其实c后缀，py后缀，txt等后缀的文件都是文本文件（不严谨地说）

我们如何去编辑呢？比如说114514.txt这个文件，我们只需要输入

```
nano 114514.txt
```

即可用nano这个编辑器程序打开一个名为114514.txt的文件，打开后的界面是这样的：

当我们修改完，并输入`Ctrl+X`退出时，它会提示你：”保存修改后的缓冲区？（回答 "N "将取消更改）。“



此时，我们输入「y」，并回车确定，它就会保存你的修改，如果你想返回去修改，只需要`Ctrl+C`即可

此时，我们对修改文本文件这个方法已经有了一定基础啦，那么我们继续讲解命令

查找一个正在运行的程序，并让其停止运行：ps、kill（用到grep过滤和找寻我们想要的）

Ps语法

```
ps [options] [--help]
```

参数:

- ps 的参数非常多, 在此仅列出几个常用的参数并大略介绍含义
- -A 列出所有的进程
- -w 显示加宽可以显示较多的资讯
- -au 显示较详细的资讯
- -aux 显示所有包含其他使用者的进程
- au(x) 输出格式 :

```
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
```

- USER: 行程拥有者
- PID: pid
- %CPU: 占用的 CPU 使用率
- %MEM: 占用的记忆体使用率
- VSZ: 占用的虚拟记忆体大小
- RSS: 占用的记忆体大小
- TTY: 终端的次要装置号码 (minor device number of tty)
- STAT: 该行程的状态:
  - D: 无法中断的休眠状态 (通常 IO 的进程)
  - R: 正在执行中
  - S: 静止状态
  - T: 暂停执行
  - Z: 不存在但暂时无法消除
  - W: 没有足够的记忆体分页可分配
  - <: 高优先序的行程
  - N: 低优先序的行程
  - L: 有记忆体分页分配并锁在记忆体内 (实时系统或捱A I/O)
- START: 行程开始时间
- TIME: 执行的时间
- COMMAND:所执行的指令

是不是眼花缭乱？Linux的真谛就是理解程序，所以很多程序都带有`-h`或者`--help`参数来使得你知道更多

我们只需要知道下面这段代码用于查找指定进程格式：

```
ps -ef | grep 进程关键字
```

例如显示 linux 的进程：

```
root@Cloud:~# ps -ef|grep nginx
root      5249  3767  0 09:12 pts/1    00:00:00 grep --color=auto nginx
```

我们就得出了这个nginx程序的id，就可以通过`kill`杀死这个进程。其实相当于Windows下的任务管理器：

```
kill -9 5249        ## 杀死一个PID为5249的进程，这里的-9请看下面的引用解释
```

> `kill -9 PID`

> `-9` 选项对应的信号是 SIGKILL，这是一个强制终止进程的信号。使用 `-9` 选项会立即终止目标进程，不会给进程机会进行任何清理或处理工作，直接强制结束进程。因此，`kill -9` 命令通常被称为“杀死”进程，是一种强制终止进程的方式。
>
> `kill -l PID`
>
> -l选项告诉kill命令用好像启动进程的用户已注销的方式结束进程。当使用该选项时，kill命令也试图杀死所留下的子进程。但这个命令也不是总能成功–或许仍然需要先手工杀死子进程，然后再杀死父进程。

此时我们再输入：

```
ps -ef | grep nginx
```

就会发现已经无法搜索到这个进程了。**什么？你一开始就搜索不到这个进程？**那也是理所当然的，因为Nginx是我个人安装来使用的，你搜索不到当然很正常，如果你能搜索到却又没使用`apt install nginx`去安装，这只能代表系统预设了Nginx这个程序。

测试网络是否可用，下载文件，重启和关机：ping、wget、reboot、shutdown

我们如何测试网络是否连通呢？baidu.com知道吧？

我们只需要：

```
root@Cloud:~# ping baidu.com
PING baidu.com (39.156.66.10) 56(84) bytes of data.
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=1 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=2 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=3 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=4 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=5 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=6 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=7 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=8 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=9 ttl=45 time=186 ms
64 bytes from 39.156.66.10 (39.156.66.10): icmp_seq=10 ttl=45 time=186 ms
^C
--- baidu.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9009ms
rtt min/avg/max/mdev = 186.183/186.293/186.372/0.390 ms
```

显示如上图，而不是`time out`即代表成功，有网络(因为这是你访问成功的证明，有延迟)，**留意画红圈的地方**，是的，这个代码会无止境的跑下去，因为我们没有给一个参数，告诉它访问多少次(发包)就停止，所以，我们要记得。

Ctrl+C在Linux下是停止当前代码的意思

既然有网络，我们来下载点东西吧，假如我们要下载的东西的地址在：https://baidu.com/000.zip，那我们就应该这样做：

```
wget https://baidu.com/000.zip
```

此时，东西就会下载过来，但是如果终端显示：

```
root@Cloud:~# wget

Command 'wget' not found, but can be installed with:

apt install wget
```

阁下又当如何应对？笨啊，**这是告诉你系统没有安装wget的意思**，所以我们就需要安装一下wget：

```
apt install wget      ## 如果是yum那就换一下呗
```

之后他问**你是否安装，输入「y」并回车即可。**

此时我们想要重启/关机Linux怎么办?

重启：

```
reboot
```

关机：

```
shutdown -h
```

> 在 Linux 中，`shutdown` 命令用于安全地关闭系统。以下是 `shutdown` 命令的一些常见参数：
>
> - `-h`：将系统关机并关闭电源。
> - `-r`：将系统重新启动。
> - `-k`：仅向系统中的所有用户发送关机警告信息，而不会实际执行关机操作。
> - `+<time>` 或 `now`：指定关机或重启的时间。例如，`+10` 表示 10 分钟后关机，`now` 表示立即执行关机或重启。
> - `--no-wall`：关闭系统时不发送警告消息给所有用户。
> - `--poweroff`：直接关闭系统电源。
> - `--reboot`：直接重新启动系统。
>
> 使用示例：
>
> - `shutdown -h now`：立即关闭系统。
> - `shutdown -r +10`：10 分钟后重新启动系统。
> - `shutdown -h 20:00`：在 20:00 关闭系统。
>
> 这些参数可以根据需要组合使用，以便安排系统的关机或重启操作。

```
root@Cloud:~# shutdown -h
Shutdown scheduled for Thu 2023-12-14 09:31:41 CST, use 'shutdown -c' to cancel.
root@Cloud:~# Connection to 167.88.182.157 closed by remote host.
Connection to 167.88.182.157 closed.
```

可以看到系统关机成功，并将我退出了ssh连接：

压缩成zip/tar，解压zip/tar：

记得啊，如果说没有这个命令，要么你输入错了，要么还没安装了，就像`unzip`有些系统没有安装上的，得自己安装一下。好了，那我们试试压缩114514这个文件夹：


```
root@Cloud:~# ls
114514  114514.txt  114.zip
```

此时，我们就已经压缩了114514目录。

如果我们需要解压就：

```
root@Cloud:~# ls
114514  114514.txt  114.zip
root@Cloud:~# rm -rf 114514                    ## 我先删除了114514这个文件夹所有内容，防止后面解压出错
root@Cloud:~# unzip 114.zip                    ## 解压114.zip
Archive:  114.zip                              ## 正在解压
   creating: 114514/000/
 extracting: 114514/000/114514.txt
root@Cloud:~# ls                               ## 可以看到解压出来，被我们删除的文件夹又回来了，证明我们解压成功了
114514  114514.txt  114.zip
root@Cloud:~#
```

那我们如何用另一种热门的解压缩呢？tar

压缩：

```
tar -czvf 114.tar.gz 114514/*
```

其中，`-c` 选项表示创建新的压缩文件，`-z` 选项表示使用 gzip 进行压缩，`-v` 选项表示显示详细信息，`-f` 选项表示指定压缩后的文件名。`114.tar.gz` 是要创建的压缩文件的名称，`114514/*` 是要压缩的目录路径。🚬

```
root@Cloud:~# tar -czvf 114.tar.gz 114514/*
114514/000/
114514/000/114514.txt
root@Cloud:~# ls
114514  114514.txt  114.tar.gz  114.zip
```

解压：

```
tar -xvf 114.tar
```

其中，`-x` 选项表示提取文件，`-v` 选项表示显示详细信息，`-f` 选项表示指定要解压缩的文件名。🚬

```
root@Cloud:~# rm -rf 114514
root@Cloud:~# ls
114514.txt  114.tar.gz  114.zip
root@Cloud:~# tar -xvf 114.tar.gz
114514/000/
114514/000/114514.txt
root@Cloud:~# ls
114514  114514.txt  114.tar.gz  114.zip
root@Cloud:~#
```

以下是 `tar` 和 `zip` 格式在解压缩方面的一些优势和劣势：

`tar` 格式的优势和劣势：

- 优势：
- 保留文件属性：`tar` 可以保留文件的权限、时间戳等属性。
- 支持压缩和非压缩：`tar` 可以单纯打包文件，也可以与其他压缩算法结合使用，例如结合 `gzip` 进行压缩。
- 跨平台支持：几乎所有绝大多数的类 Unix 操作系统都原生支持 `tar` 格式。
- 劣势：
- 压缩率相对较低：`tar` 单独打包文件时，压缩率一般较低。

`zip` 格式的优势和劣势：

- 优势：
- 独立的压缩格式：`zip` 可以直接进行压缩和打包，不需要依赖其他工具。
- 压缩率较高：相比较于 `tar`，`zip` 格式的压缩率较高。
- 劣势：
- 不保留文件属性：在某些情况下，`zip` 格式可能无法保留文件属性，如权限和时间戳。
- 可移植性较差：在某些非 Windows 系统中，可能需要额外安装支持 `zip` 格式的软件。

综上所述，`tar` 格式在保留文件属性和跨平台支持方面具有优势，同时可以与压缩算法结合使用；而 `zip` 格式在压缩率方面较优，但在保留文件属性和可移植性上存在一些劣势。根据具体的应用场景和需求，可以选择适合的格式进行打包和解压缩。

目录权限和目录所有者：chmod、chown（这个一般在Web服务器见的比较多，还有FTP服务器等，所以这里也不作太深入的介绍用户组和权限，因为一般的使用很少用得上，下面深入浅出讲一下）

`chmod` 命令 - 修改文件或目录的权限

- **概念和用途：** `chmod` 命令用于修改文件或目录的权限。文件权限包括读取、写入和执行权限，每个文件有属主、属组和其他用户的权限。`chmod` 命令可以更改这些权限。
- **使用方法：** `chmod` 命令的基本语法如下：

  ```
  chmod [选项] 模式 文件名
  ```

  其中，`选项` 是一些控制权限修改行为的参数，`模式` 是用数字或符号表示的权限模式，`文件名` 是要修改权限的文件名。例如：

  ```
  chmod 755 file.txt
  ```

  上述命令将 file.txt 文件的权限设置为 `rwxr-xr-x`。

`chown` 命令 - 修改文件或目录的所有者和所属组

- **概念和用途：** `chown` 命令用于修改文件或目录的所有者和所属组。在某些情况下，只有文件的所有者或者超级用户才能修改文件的属主或所属组。
- **使用方法：** `chown` 命令的基本语法如下：

  ```
  chown [选项] 新所有者:新所属组 文件名
  ```

  其中，`选项` 是一些控制权限修改行为的参数，`新所有者` 是指定的新的所有者，`新所属组` 是指定的新的所属组，`文件名` 是要修改权限的文件名。例如：

  ```
  chown user1:group1 file.txt
  ```

  上述命令将 file.txt 文件的所有者设置为 user1，所属组设置为 group1。

异同点

- `chmod` 主要用于修改文件或目录的权限（读、写、执行权限），而 `chown` 用于修改文件或目录的所有者和所属组。
- `chmod` 的参数是权限模式，可以使用数字或符号表示，而 `chown` 的参数是新的所有者和所属组。
- 两者都是用于管理文件和目录的权限，但是作用范围不同，`chmod` 只修改权限，`chown` 只修改所有者和所属组。

绝杀 - 搜寻过滤器：grep

因为grep能做的事太广泛了，下面我直接引用生成式答案：

`grep` 命令用于在文件中搜索指定的模式，并将包含匹配文本行打印输出。

1. 搜索包含指定关键词的文件：

```
grep "keyword" filename
```

这将在 `filename` 文件中搜索包含指定关键词 "keyword" 的所有行，并将它们打印输出。

2. 递归搜索指定目录下所有文件中的内容：

```
grep -r "keyword" /path/to/directory
```

通过使用 `-r` 选项，`grep` 将递归搜索指定目录下所有文件中包含指定关键词 "keyword" 的行，并将它们打印输出。

3. 显示匹配文本行之前/之后的内容：

```
grep -A 3 "keyword" filename  # 显示匹配行之后的 3 行内容
grep -B 2 "keyword" filename  # 显示匹配行之前的 2 行内容
```

`-A` 选项用于显示匹配行之后的内容，`-B` 选项用于显示匹配行之前的内容。

4. 忽略大小写进行搜索：

```
grep -i "keyword" filename
```

通过 `-i` 选项忽略大小写，`grep` 将匹配大小写不敏感的关键词 "keyword"。

5. 统计匹配行的数量：

```
grep -c "keyword" filename
```

`-c` 选项用于统计文件中包含指定关键词的行数。

这些是 `grep` 命令的一些常见应用实例，它可以帮助用户在文本文件中快速搜索和定位指定的内容。

最后：

目前教程就到此结束，前前后后写了两天的教程暂时也画上了句号。希望你可以在这个教程里可以对Linux有初步的了解并入门它，由于是个人赶制，所以可能会有不少错误纰漏，还有不严谨和做得不好的地方，敬请原谅。