Linux命令行基础

By 陆逸文

0 概述

什么是Linux? Linux和Windows, Mac一样，是一种计算机操作系统[1]。Linux有各种各样的「发行版」，它们拥有相同的Linux内核，以及不尽相同的预装软件和软件包管理方式。我们所耳熟能详的Ubuntu就是诸多Linux发行版中的一种。

为什么学习Linux命令行? Linux系统具有自由、可定制、稳定等特点，因此成为服务器领域的主流，科协网站、新生C平台都是在Linux系统上运行的。抛开这些不谈，同学们日后做项目、做科研的时候也肯定会有跟Linux命令行打交道的时候。另外，熟悉Linux命令行对在Windows和Mac下做开发也很有帮助，毕竟原理是共通的。

本教程的内容和目的 本教程将带领大家登录一台能用的Linux机器、熟悉一些最常用的命令、掌握在Linux命令行下工作的基本方法。Linux下的命令浩如烟海，全部介绍不仅枯燥而且没有可能，用到时查阅帮助文档即可。本教程的主要目的是让大家在看到「黑框框」时不再感到束手无策 :-)

注意，本教程介绍的是「命令行」，与之相对的就是「图形界面」啦~ 事实上，Ubuntu等发行版都提供了类似Windows, Mac的图形化桌面环境。如果觉得从命令行上手曲线略陡的话，装个桌面环境玩玩也是极好的，但那些就没必要重点介绍了。

Tip for Mac users: Mac和Linux都属于类Unix系统，可谓「本是同根生」，因此本教程介绍的大部分内容对Mac命令行也适用或者仅仅存在细微的差别。如果你还没有使用过Mac下的命令行，不妨打开「终端」应用体验一下，并安装Homebrew (https://brew.sh)，该工具类似下文介绍的`apt`，可谓Mac下做开发的必备神器。

1 登录一台Linux机器

方法一：安装虚拟机

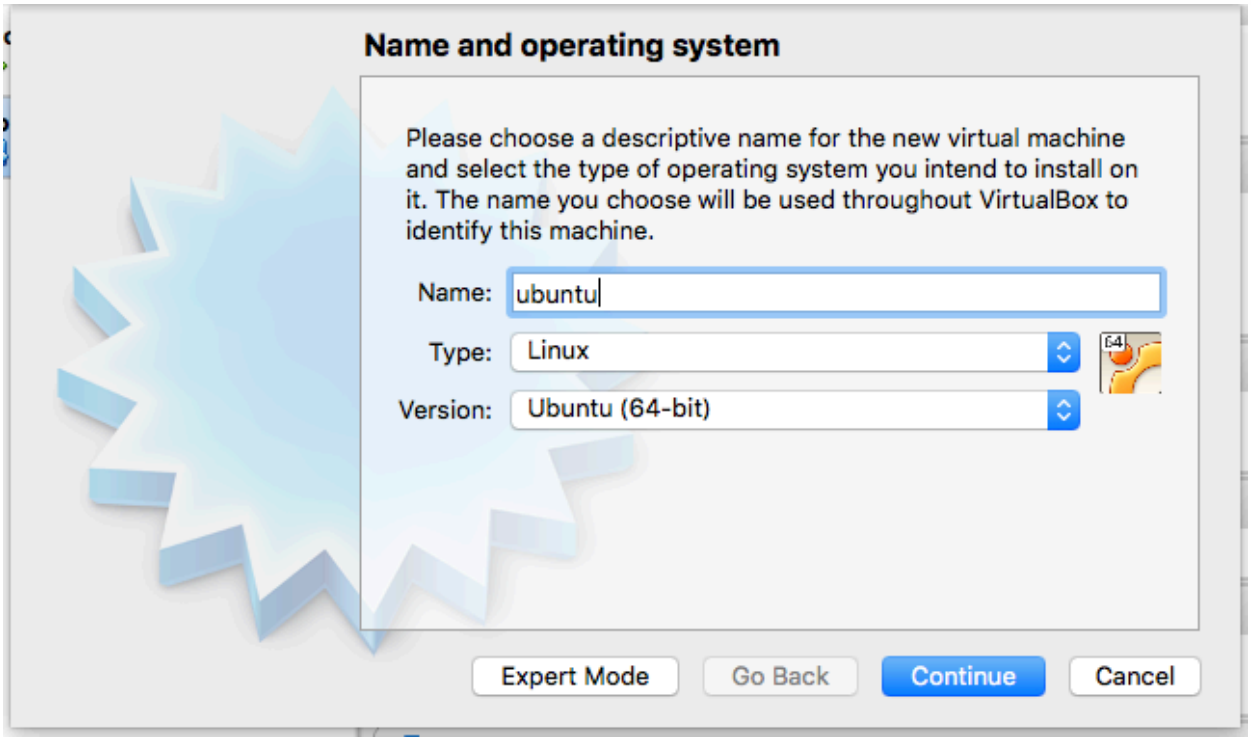
你需要准备：

- 一个虚拟机软件
- 一个系统镜像（相当于操作系统安装盘）
- 充足的硬盘空间和内存 :-)

常用的虚拟机软件包括VMWare、VirtualBox等，直接下载安装就好。各种常用Linux发行版的镜像都可以在清华TUNA镜像站(https://mirrors.tuna.tsinghua.edu.cn)下载（校内免流量哦~）。推荐安装Ubuntu 17.04 (https://mirrors.tuna.tsinghua.edu.cn/ubuntu-releases/17.04/)；32位(i386)或64位(amd64)版本皆可，推荐64位（然而个别旧电脑可能需要在BIOS里改一下设置才能运行64位虚拟机...）；server和desktop版就本教程的目的而言皆可，主要区别是server版默认不带图形界面，装上就只能看到黑框框了😂。选好版本以后点击下载对应的后缀为 iso 的文件。

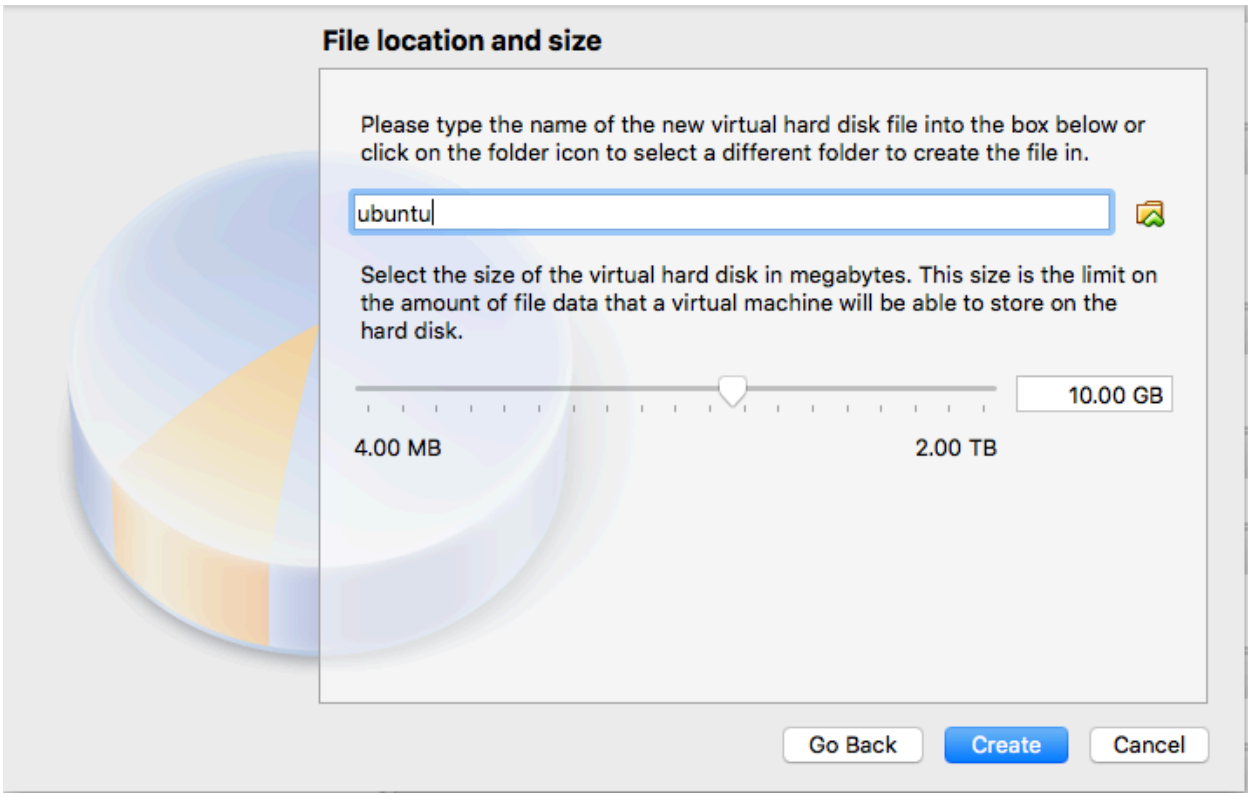
下面以VirtualBox和Ubuntu 17.04 Desktop为例简单演示虚拟机的安装：

打开VirtualBox，点击"New"按钮，名称输入Ubuntu，VirtualBox会自动选择配置：



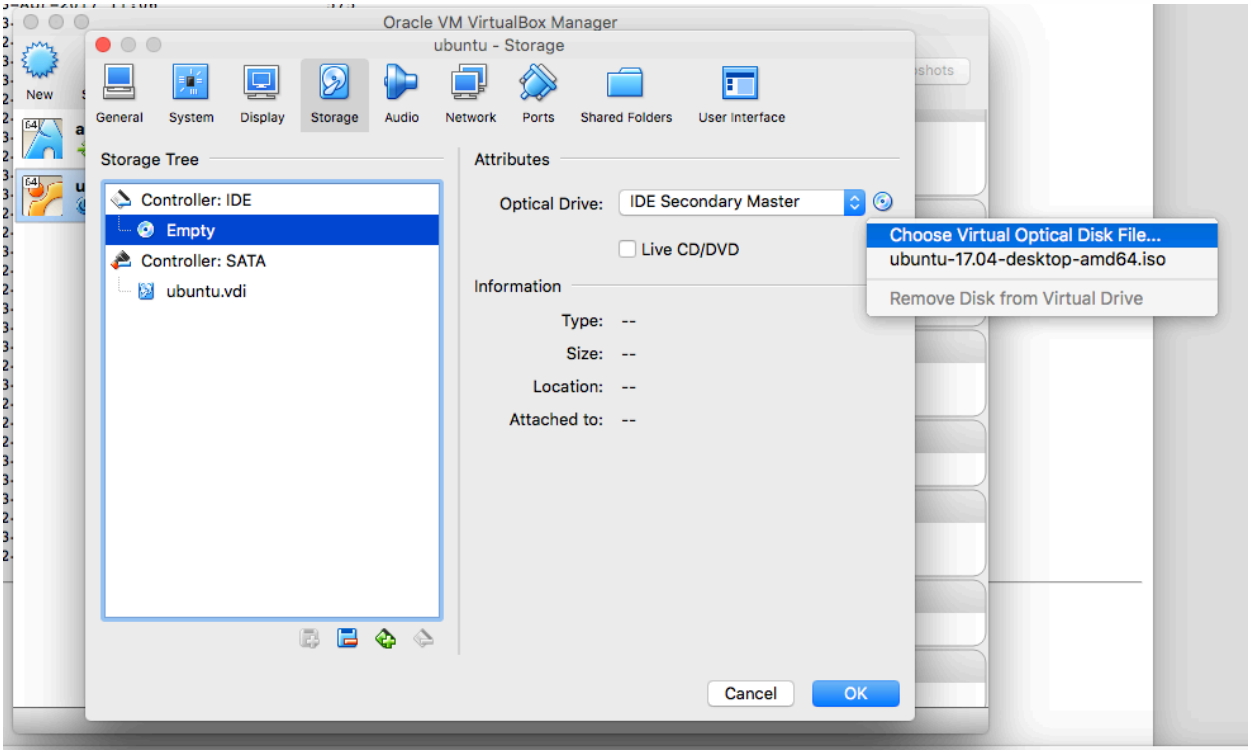
下一步，内存大小：推荐的1024MB应该是一个下限，如果电脑内存比较充裕的话建议多给一些，比如2048MB。

下一步，硬盘：选择创建虚拟硬盘，一直继续到这一步：



同样建议多给一点，比如20GB。因为之前选择了「动态分配」的方式，这里的输入的硬盘大小只是一个上限值，如果用不到这么多的话也不会占用实际的硬盘空间的。

创建完虚拟机后，选择刚刚创建的虚拟机，点击Settings > Storage > Controller:IDE下的Empty，找到光盘图标，在"Choose Virtual Optical Disk File"中选择刚才下载的iso文件。这个操作相当于向虚拟机插入安装光盘。



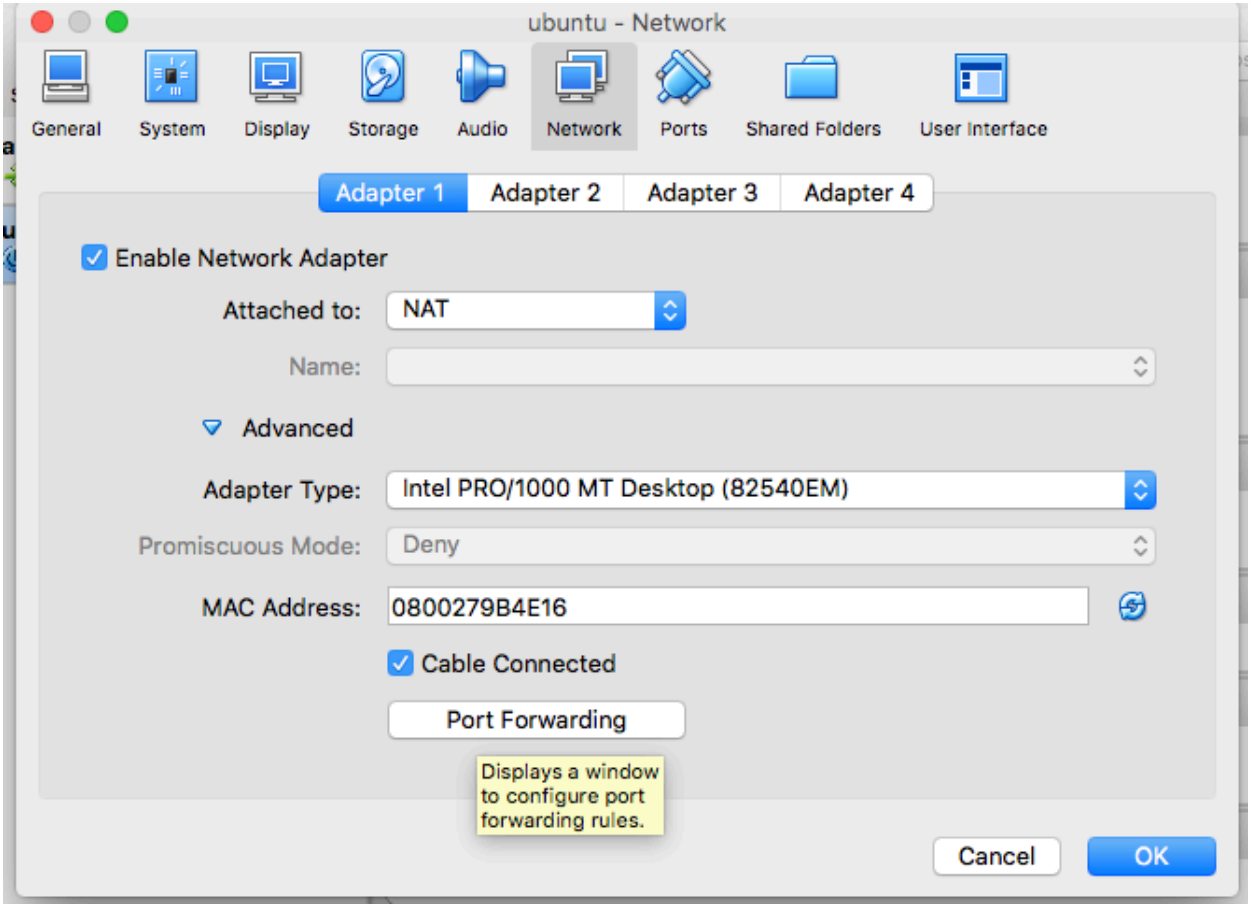
接下来开机，按照提示一步步操作就行了。安装完成后回到这个Storage界面，在光盘图标的菜单下选择"Remove Disk from Virtual Drive"（弹出光盘）。

开机之后，按Ctrl+Alt+T，打开终端，就可以进行命令行操作了 :-)

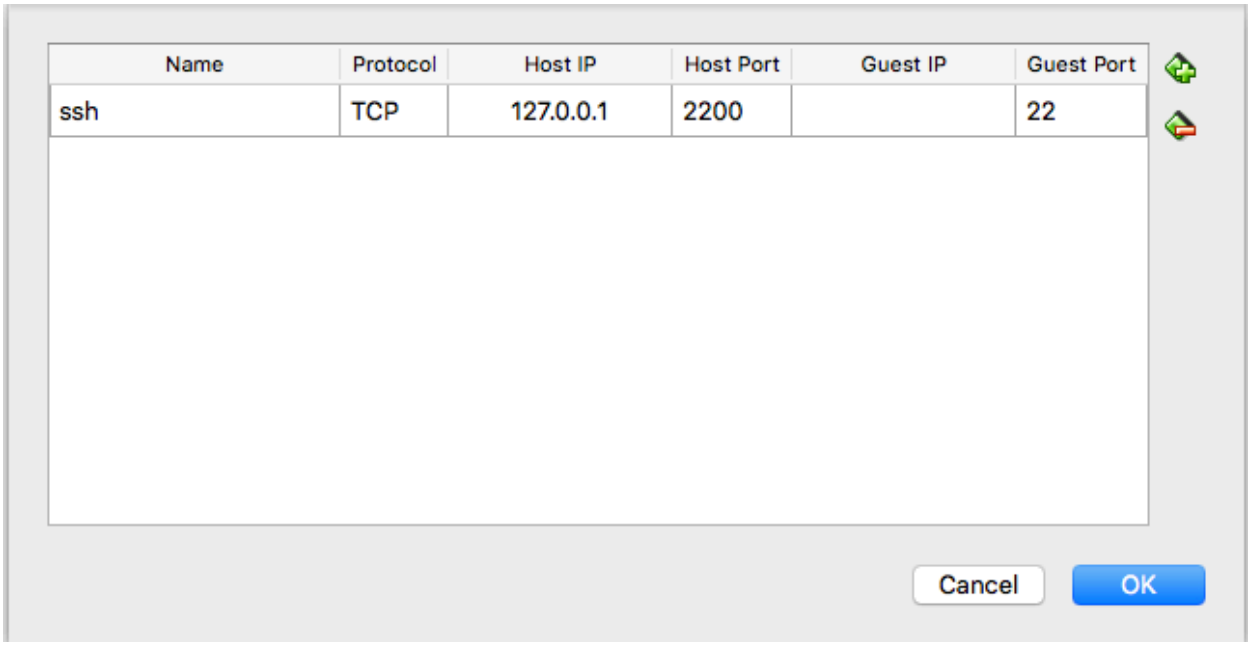
设置远程登录（可选）

建议先跳过这一部分，等完成整个教程后有兴趣再回过头来设置。

先设置端口映射：在虚拟机设置里进入Network > Port Forwarding：



新建一条规则，如图所示：



其中名称可以任取，Host Port也可在1024~65535的整数中任取（只要没有其他服务占用所选的端口即可），下文中默认为2200端口，如果选择了其他的端口请自行代入。

这一步操作的目的是将虚拟机(Guest)的22端口（也就是远程登录用的ssh服务所使用的端口）映射到真机(Host)的2200端口，这样在真机上访问127.0.0.1:2200[2]就可以「远程」登录虚拟机了。（如果这一段暂时没看懂也不要紧）

打开虚拟机，安装`openssh-server`：

`sudo apt install openssh-server`

现在，在真机中打开先前安装的Git Bash，输入`ssh yourname@127.0.0.1 -p 2200`，（其中`yourname`是你在虚拟机中的用户名，如果虚拟机和真机中用户名一致的话`yourname@`也可以省略不打）就可以从真机中操控虚拟机的命令行了~

如果先前在配置Git时生成了SSH Key的话，你还可以在Git Bash中输入`ssh-copy-id yourname@127.0.0.1 -p 2200`，以后再登录虚拟机就不需要密码了。

事实上，VirtualBox还允许在主机中通过命令在后台启动虚拟机，而不显示图形界面（但仍然可以通过ssh登录）。感兴趣的同学请自行学习（搜索关键词：virtualbox headless）。

关于虚拟机的更多（可选）

近年一些比较流行的虚拟机自动化部署工具大大简化了配置虚拟机的流程，如Vagrant, Docker等(Docker严格来说不是虚拟机，而是容器技术)。虚拟化技术的应用可不仅仅是在Windows下跑Ubuntu那么简单，它使Web应用的大规模部署变得更加简单。感兴趣的同学可以自行查阅资料了解（关键词：Vagrant, Docker, PaaS, SaaS, IaaS）

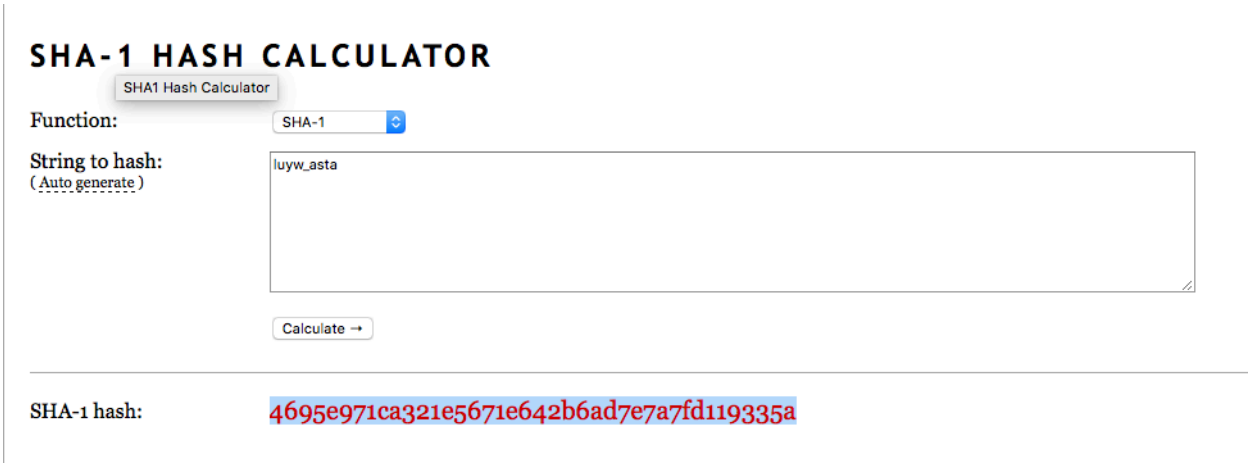
方法二：安装双系统

安装双系统较安装虚拟机要更麻烦一些，需要多出制作启动盘、设置分区、设置启动引导等环节，且根据电脑的不同可能会遇到一些奇奇怪怪的问题。有兴趣的同学请自行参阅网上资料。（**警告：请备份好数据，并做好在一些莫名其妙的bug上浪费很多时间的心理准备...**）

方法三：远程登录科协服务器

人生苦短，何必浪费时间在配置环境上！我们在科协服务器上为每位参加讲座的同学创建了帐号，如果暂时不想在本地装Linux系统的话可以使用。（如果不在讲座群里的同学需要帐号的话请私戳我）

打开先前安装的Git Bash, 输入`ssh username@daasta.cn`，其中`username`是你的姓的全拼加上名的各个字的首字母，e.g. 陆逸文 -> `luyw`。密码可以在这个网站上计算得到：http://www.md5calc.com。Function选择SHA-1，String to hash输入你的用户名后面加上`_asta`，点击Calculate后得到的一串东西就是你的初始密码。



登录上去以后，输入`passwd`命令可以修改密码。

如果先前在配置Git时生成了SSH Key的话，还可以在Git Bash中输入`ssh-copy-id username@daasta.cn`，以后再登录服务器就不需要输入密码了。

Have fun :-)

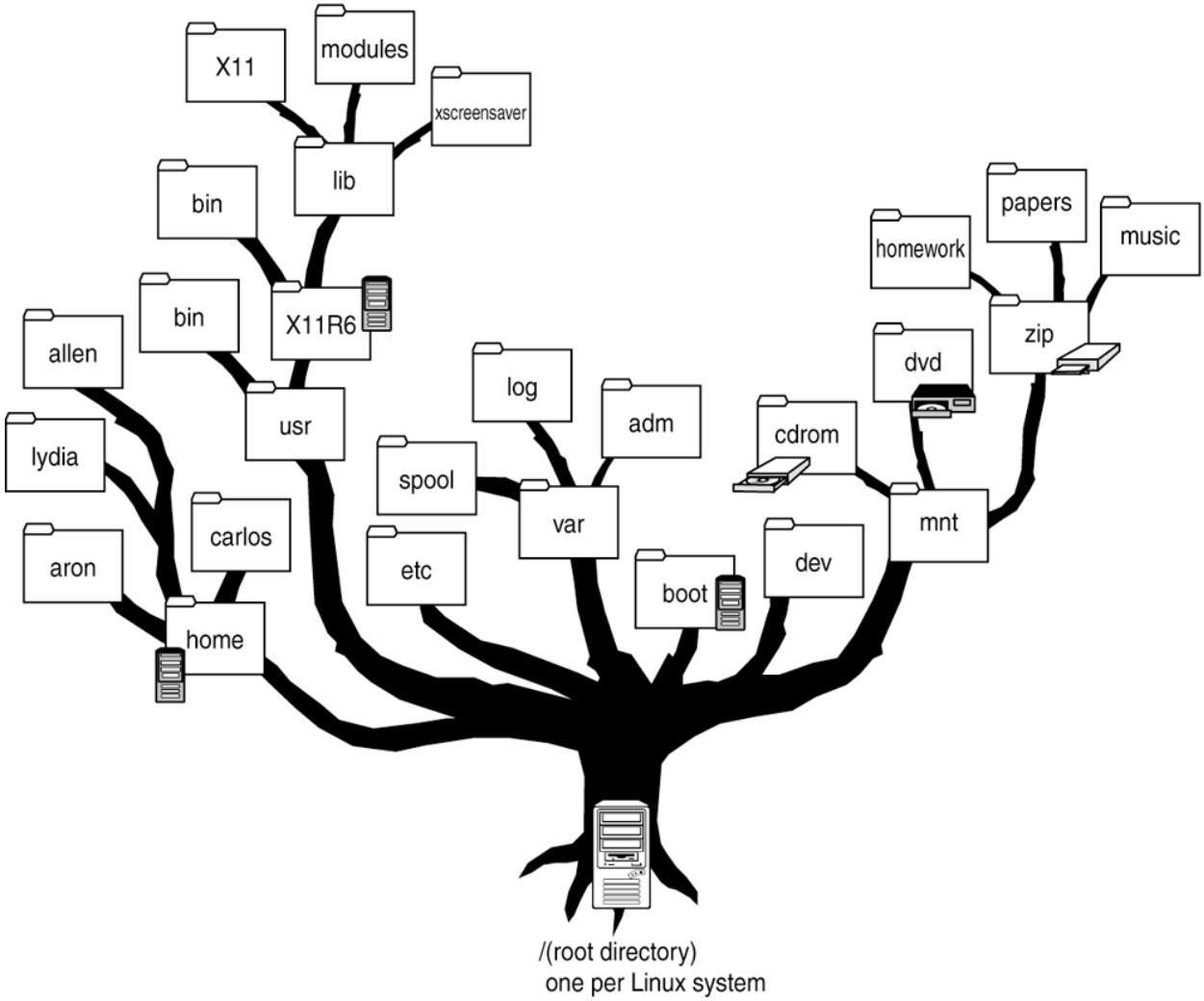
2 文件管理

常用命令一览

命令	全称	解释
`cd`	change directory	切换目录
`ls`	list	罗列当前目录下的文件/子目录
`touch`	touch	新建文件/将已经存在的修改时间更新到现在
`mkdir`	make directory	新建目录
`cp`	copy	复制
`mv`	move	移动/重命名
`rm`	remove	删除
`cat`	concatenate（为什么？）	显示文件内容
`less`	less	对于较长的文件，显示文件的头部，可用键盘方向键向下滚动

Linux的目录结构

Linux文件系统是一个树状的结构。表示一个路径时，不同级别的节点之间用 `/` 字符分隔。比如在下图中，`X11`这个节点的路径可以表示为`/usr/X11R6/lib/X11`。



接下来我们看看自己电脑的目录结构长什么样。先切换到树根：`cd /`。输入`ls`即可列举出树根的所有直接子节点。以科协服务器为例：

```
➜  / ls -l
total 84
drwxr-xr-x   2 root root  4096 Apr 28 14:17 bin
drwxr-xr-x   3 root root  4096 Feb 20 12:28 boot
drwxr-xr-x  20 root root  4160 Apr 24 17:54 dev
drwxr-xr-x  98 root root  4096 Jul 10 00:49 etc
drwxr-xr-x  24 root root  4096 Jul 10 00:45 home
lrwxrwxrwx   1 root root    32 Feb 20 12:27 initrd.img -> boot/initrd.img-4.4.0-63-
generic
lrwxrwxrwx   1 root root    32 Feb 20 12:18 initrd.img.old -> boot/initrd.img-
4.4.0-31-generic
drwxr-xr-x  20 root root  4096 Feb 20 12:29 lib
drwxr-xr-x   2 root root  4096 Feb 20 12:29 lib64
drwx------   2 root root 16384 Feb 20 12:17 lost+found
drwxr-xr-x   4 root root  4096 Feb 20 12:17 media
drwxr-xr-x   2 root root  4096 Jul 20  2016 mnt
drwxr-xr-x   2 root root  4096 Jul 20  2016 opt
dr-xr-xr-x 130 root root     0 Apr 24 17:54 proc
drwx------   5 root root  4096 Apr 24 18:10 root
drwxr-xr-x  22 root root   740 Jul 10 15:43 run
drwxr-xr-x   2 root root 12288 Feb 20 12:26 sbin
drwxr-xr-x   2 root root  4096 Jul 20  2016 srv
dr-xr-xr-x  13 root root     0 Jul 10 15:43 sys
drwxrwxrwt   7 root root  4096 Jul 10 15:45 tmp
drwxr-xr-x  10 root root  4096 Feb 20 12:17 usr
drwxr-xr-x  12 root root  4096 Apr 24 18:17 var
lrwxrwxrwx   1 root root    29 Feb 20 12:27 vmlinuz -> boot/vmlinuz-4.4.0-63-
generic
lrwxrwxrwx   1 root root    29 Feb 20 12:18 vmlinuz.old -> boot/vmlinuz-4.4.0-31-
generic
```

看起来很乱？不要紧，现在只要了解以下几个目录名称的含义就可以了：

`/bin`

binary的缩写，系统核心的一些可执行文件。不妨看看里面的内容：

```
➜  / ls /bin
bash            gunzip          ntfs-3g.secaudit  su
bunzip2         gzexe           ntfs-3g.usermap   sync
busybox         gzip            ntfscat           systemctl
bzcat           hostname        ntfscluster       systemd
bzcmp           ip              ntfscmp           systemd-ask-password
bzdiff          journalctl      ntfsfallocate     systemd-escape
bzegrep         kbd_mode        ntfsfix           systemd-hwdb
bzexe           kill            ntfsinfo          systemd-inhibit
bzfgrep         kmod            ntfsls            systemd-machine-id-setup
bzgrep          less            ntfsmove          systemd-notify
bzip2           lessecho        ntfstruncate      systemd-tmpfiles
bzip2recover    lessfile        ntfswipe          systemd-tty-ask-password-agent
bzless          lesskey         open              tailf
bzmore          lesspipe        openvt            tar
cat             ln              pax               tempfile
chgrp           loadkeys        paxcpio           touch
chmod           login           paxtar            true
chown           loginctl        pidof             udevadm
chvt            lowntfs-3g      ping              ulockmgr_server
cp              ls              ping6             umount
cpio            lsblk           plymouth          uname
dash            lsmod           ps                uncompress
date            mkdir           pwd               unicode_start
dd              mknod           rbash             vdir
df              mktemp          readlink          wdctl
dir             more            red               which
dmesg           mount           rm                whiptail
dnsdomainname   mountpoint      rmdir             ypdomainname
domainname      mt              rnano             zcat
dumpkeys        mt-gnu          run-parts         zcmp
echo            mv              rzsh              zdiff
ed              nano            sed               zegrep
egrep           nc              setfont           zfgrep
false           nc.openbsd      setupcon          zforce
fgconsole       netcat          sh                zgrep
fgrep           netstat         sh.distrib        zless
findmnt         networkctl      sleep             zmore
fuser           nisdomainname   ss                znew
fusermount      ntfs-3g         static-sh         zsh
grep            ntfs-3g.probe   stty              zsh5
```

看到了熟悉的面孔？对，之前在「常用命令一览」中介绍的每一个命令都是`/bin`下的一个文件。它们相当于是Windows下的`.exe`文件，只是Linux下的可执行文件没有专门的后缀罢了。

`/etc`

etc跟平时英语中写的etc.是来自同一拉丁文单词et cetera，可以理解为「一些零碎的东西」。这里是配置文件集中存放的地方。可以认为类似于Windows下的注册表。与Windows不同的是，这里的所有配置都是以纯文本形式存储的。这是Unix的"KISS (Keep It Simple, Stupid)"哲学的一个典型体现，客观来说跟Windows的方式各有千秋，自己慢慢体会吧~

`/home`

顾名思义，用户文件存放的位置。看一下它的内容就很明白了：

```
➜  / ls -l /home
total 88
drwxr-xr-x 10 asta    asta     4096 Jul 10 00:23 asta
drwxr-xr-x  2 chengxt tutorial 4096 Jul 10 00:45 chengxt
drwxr-xr-x  2 guopk   tutorial 4096 Jul 10 00:45 guopk
drwxr-xr-x  2 haocf   tutorial 4096 Jul 10 00:45 haocf
drwxr-xr-x  3 huangr  tutorial 4096 Jul 10 10:26 huangr
drwxr-xr-x  2 huangsc tutorial 4096 Jul 10 00:45 huangsc
drwxr-xr-x  2 kangwx  tutorial 4096 Jul 10 00:45 kangwx
drwxr-xr-x  2 liz     tutorial 4096 Jul 10 00:45 liz
drwxr-xr-x  2 louzq   tutorial 4096 Jul 10 00:45 louzq
drwxr-xr-x  2 luyf    tutorial 4096 Jul 10 00:45 luyf
drwxr-xr-x  7 luyw    asta     4096 Jul 10 16:21 luyw
drwxr-xr-x  2 sunzl   tutorial 4096 Jul 10 00:45 sunzl
drwxr-xr-x  2 tangxw  tutorial 4096 Jul 10 00:45 tangxw
drwxr-xr-x  3 taohz   tutorial 4096 Jul 10 16:13 taohz
drwxr-xr-x  3 tiany   tutorial 4096 Jul 10 00:46 tiany
drwxr-xr-x  2 wangqt  tutorial 4096 Jul 10 00:45 wangqt
drwxr-xr-x  2 wangyh  tutorial 4096 Jul 10 00:45 wangyh
drwxr-xr-x  2 weiyx   tutorial 4096 Jul 10 00:45 weiyx
drwxr-xr-x  2 wut     tutorial 4096 Jul 10 00:45 wut
drwxr-xr-x  2 xiaoh   tutorial 4096 Jul 10 00:45 xiaoh
drwxr-xr-x  2 xiazf   tutorial 4096 Jul 10 00:45 xiazf
drwxr-xr-x  2 zhongqy tutorial 4096 Jul 10 00:45 zhongqy
```

每个用户的用户名都对应于`/home`下的一个子目录。找到属于自己的目录就可以在里面存放文件了。

`/usr`

别误会了，`usr`不是"user"，它是Unix System Resources的缩写。部分系统自带的和用户安装的各种软件和库都会储存在这里。比如git的完整路径是`/usr/bin/git`。

`/var`

表示"variable"，「可变」之义。系统的临时文件等等都会放在这里。

思考 Linux这样的目录结构有什么好处呢?

一些特殊的目录名称

- `~`: 表示「home下属于当前用户的目录」。比如说用我的帐号(`luyw`)登录时，`~`就相当

于`/home/luyw`。

- `..`: 表示「上一级目录」。`cd ..`相当于在图形界面中点「向上」。
- `.`: 表示「当前目录」。为什么需要这种看似啥都不做的东西呢？保持好奇~

Tip: 这些特殊的目录名称都可以跟`/`组合使用哦~

练习 熟练掌握「常用命令一览」中各命令的使用方法。

Tip: 不清楚一个命令的用法的时候，可以通过`man`(manual)命令获得帮助。比如通过`man cp`可以查看`cp`命令的帮助文档。

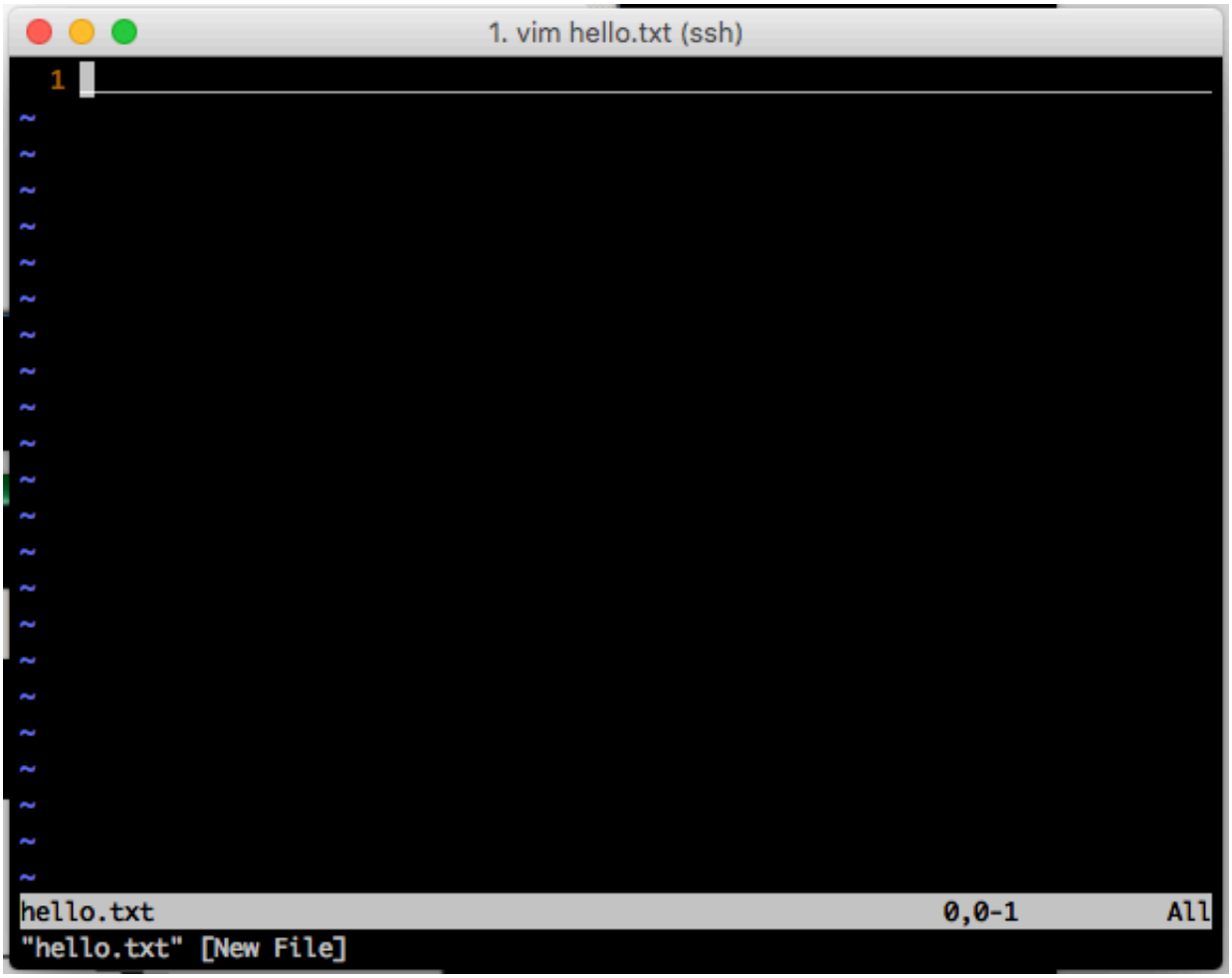
Tip: 如果对文件夹整体操作遇到问题，尝试`-r`(recursive)选项。

3 使用vi/vim文本编辑器

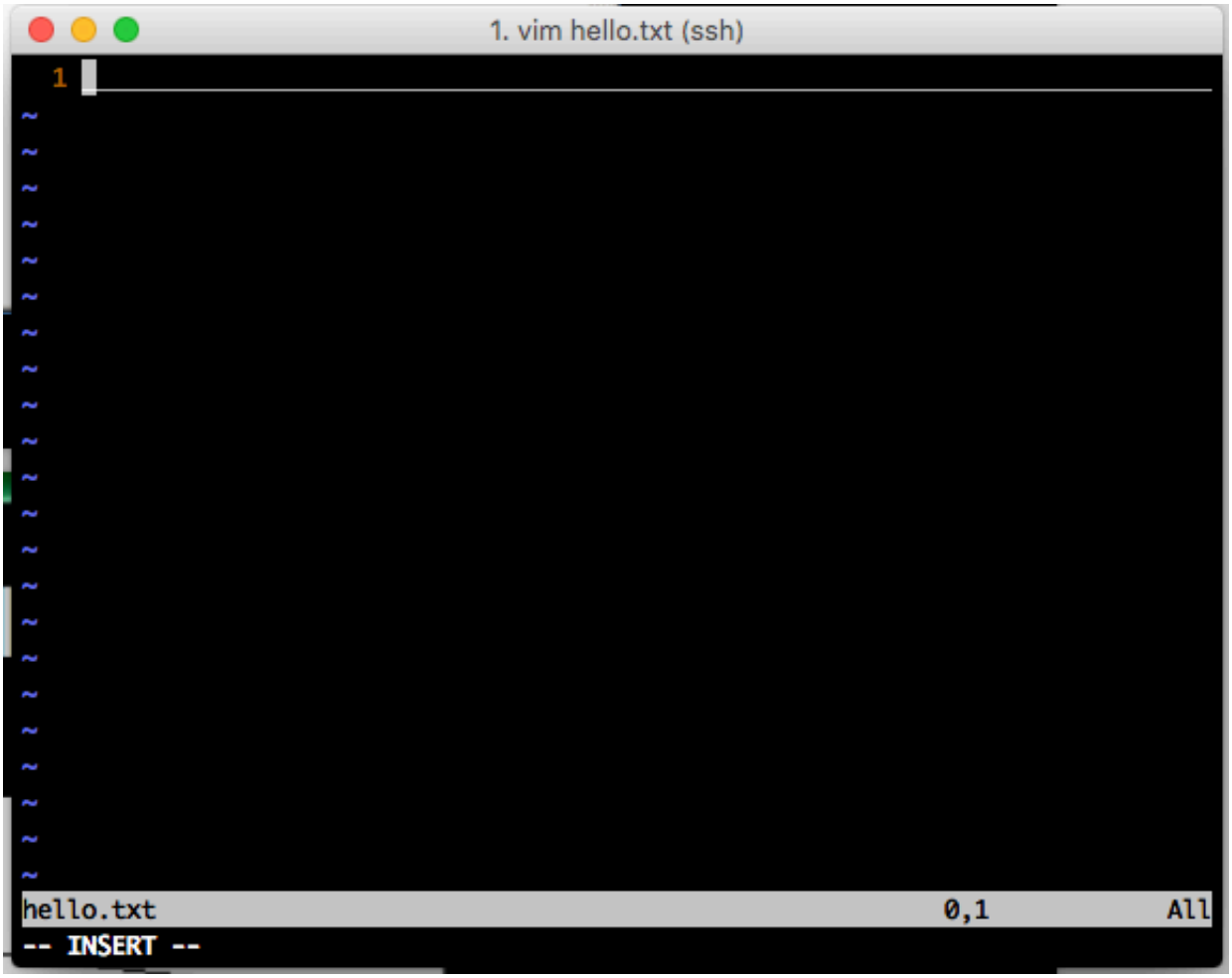
vi或vim(全程vi improved)是一款著名的命令行文本编辑器。vim固然强大，但纯命令行操作毕竟不够方便，大家以后也未必会经常使用vim。为何要学习这一套有些晦涩的操作呢?

- 如果要在一台没有图形界面的服务器上编辑文本，vim几乎是唯一的选择。
- 掌握vim的键位对提高写码效率很有帮助。事实上，大部分主流的文本编辑器和IDE都支持vim插件 :-)

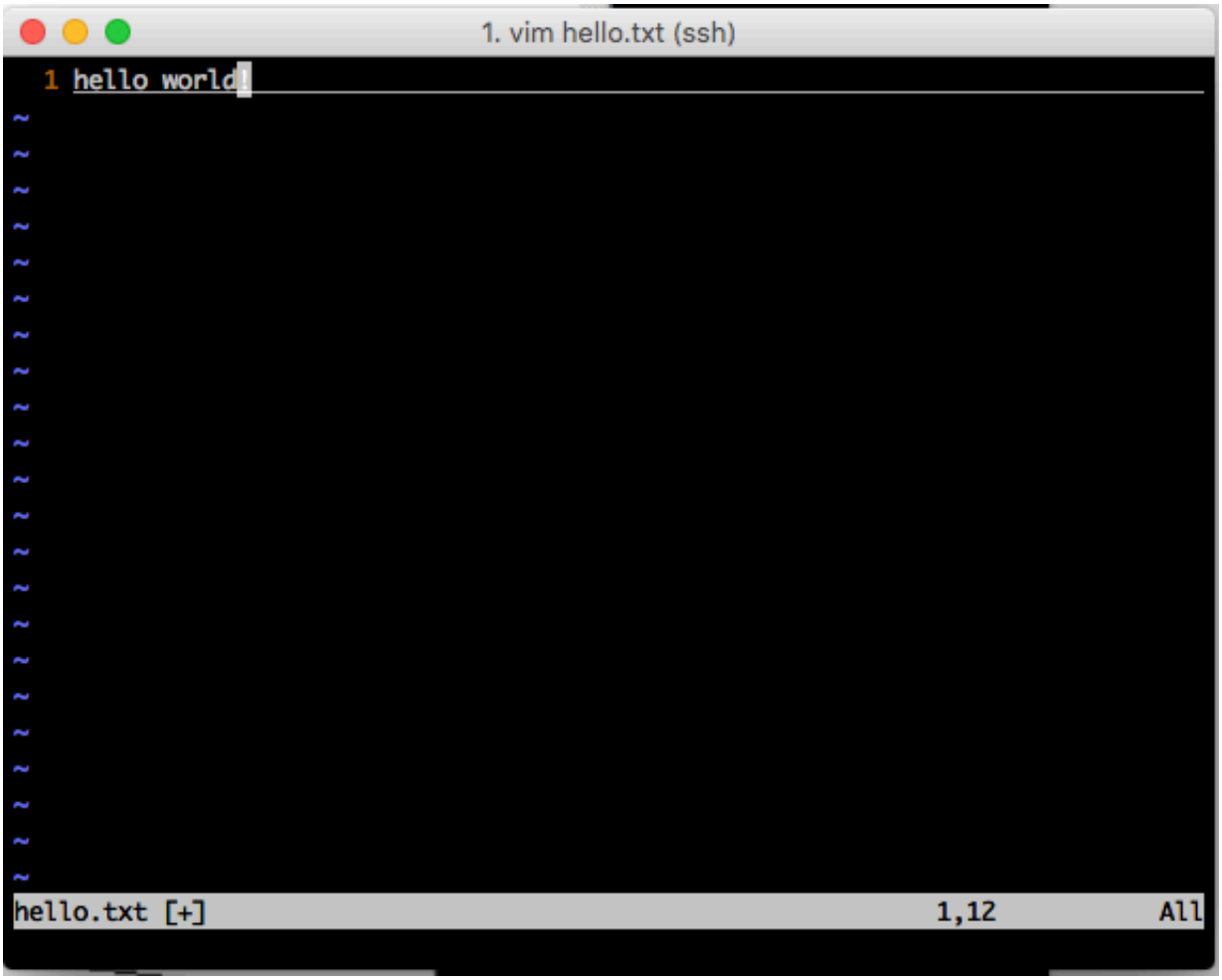
切换到一个合适的目录，输入`vim hello.txt`（无需事先创建，如果用vim打开一个不存在的文件那自动就创建了），应该可以看到类似于这样的界面：



在键盘上点击`i`，就变成了这样：



注意屏幕下方出现了`-- INSERT --`字样。现在，在键盘上输入的任意文字都会出现在屏幕上了。完成输入后，在键盘上点击`ESC`键，这时屏幕下方的`-- INSERT --`字样应该消失了：



输入`:wq`，敲击回车，保存并退出。

vim的高效性体现在它多样的编辑命令上。vim自带一个交互式的教程，请同学们输入`vimtutor`命令自行学习。

另外，vim的可定制性非常强，插件数量繁多，这里不再展开介绍。

练习 输入`vimtutor`命令，熟悉vim的基本操作。

4 权限

思考 服务器上那么多用户，我们是不是可以随意查看和修改其他用户目录下的文件呢?

试试看 编辑一个系统目录下的文件，比如`vim /etc/hosts`，进行一些修改，可以正常保存吗?

在Linux中，每个文件/目录都有一个由三个三位二进制数组成的权限。每个二进制数的三位分别表示读、写、执行；三个二进制数分别表示所有者的权限、跟所有者同组的其他用户的权限、其他人的权限。

在`ls -l`的输出中，每行最开头的那一串东西表示的就是权限。举个栗子：`drwxr-xr-x`，开头的`d`表示directory（这是一个目录的权限；如果是文件的话会显示为`-`）。后面的九个字母可以三位一组分成三组来看。所有者的权限是`rwx`，表示可读、可写、可执行；同组其他用户和其他人的权限都是`r-x`，表示可读、不可写、可执行。这个目录的权限也可以用数字表示为`755`（7和5分别对应于二进制数`111`和`101`）。

通过`chmod`命令可以修改权限。指定权限的方式既可以是通过三位数字，也可以通过字母添加或减去某一项权限。

练习 新建一个文件或操作一个已有的文件（假设叫`hello.txt`），尝试以下命令：

- `chmod 755 hello.txt`
- `chmod o-r hello.txt`
- `chmod g+w hello.txt`
- `chmod -x hello.txt`

每次运行完一条命令，运行`ls -l hello.txt`观察权限的变化。

练习 新建一个目录（假设叫`hello`），并在目录中新建一个文件（假设叫`hello/hello.txt`），尝试以下命令：

- `chmod 777 hello`
- `chmod -R 777 hello`（注意对比这两条命令；这里的`-R`是recursive之意）
- `chmod -R -x hello`

每次运行完一条命令，运行`ls -l | grep hello`和`ls -l hello`分别观察目录和文件的权限变化。在运行完最后一条命令后，尝试`cd hello`，你发现了什么？

关于「运行」权限

读写权限容易理解，但到底什么是「运行」权限呢？它对文件和目录的含义是不同的。

创建一个名为`hello.sh`的文件，内容如下：

```
#!/bin/bash
echo hello
```

这是一个最简单的Shell脚本，`echo`类似C语言中的`printf`，这个脚本的作用就是输出"hello"。

练习 依次执行以下命令：

- `./hello.sh`
- `bash hello.sh`
- `chmod +x hello.sh`
- `./hello.sh`

可以观察到，加了运行权限后，脚本就可以直接用`./`调用了。

思考 比较两种调用方式，`./hello.sh`较`bash hello.sh`有什么好处？

对于目录，运行权限等同于「cd到这个目录的权限」。在上上个练习中，取消运行权限后无法cd进`hello`就是这个原因。

Root

用过Android手机的同学可能听说过"Root"这个说法。事实上，在Linux系统中，Root用户是一个神一样的存在，它不受读写权限的约束，可以读写系统中的任何文件。用户如果希望以Root身份运行一条命令，可以在原来的命令前面加上`sudo`，这就相当于Windows下的「以管理员身份运行」。一些涉及到修改系统设置的操作都需要Root权限。（同学们可以试一下在科协服务器上使用sudo，会发现无法提权，这是因为我没有把你们加到sudo用户组里，所以我就不怕你们搞破坏啦 :-p）

练习 递归修改`/home`下自己的用户目录的权限，让其他人不能访问。

5 搜索路径

我们前面提到，`cp`的完整路径是`/bin/cp`，而`git`的完整路径是`/usr/bin/git`；其他命令的完整路径也可以通过`which`命令查看。那么系统是如何找到这些分散在各个不同路径下的命令的呢?

答案是一个叫`PATH`的环境变量，打印它看看：

```
➜  ~ echo $PATH
/usr/local/sbin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
```

`PATH`由一系列路径构成，路径与路径之间用冒号分隔。在运行一个命令是，系统会依次在这些路径中进行搜索。

我们可以通过Shell中的赋值语句对`PATH`进行修改。比如将`/home/luyw`添加到`PATH`的最后：

```
➜  ~ PATH=$PATH:/home/luyw
➜  ~ echo $PATH
/usr/local/sbin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/home/luyw
```

这个修改是临时的，退出当前Shell以后就失效了。要使修改永久有效，可以将`PATH=$PATH:/home/luyw`这句话添加到`~/.bashrc`中。

之前我们关于`.`和`./`的疑惑也得到了解答。比如当前目录下有一个叫`hello.sh`的脚本（已赋予运行权限），如果直接运行`hello.sh`，因为当前路径不在`PATH`中，所以系统是无法找到我们要运行的脚本的；`./hello.sh`相当于是告诉了系统脚本所在的位置。

6 管理软件包

几乎所有常见Linux发行版都包含*包管理器*；常用的软件和库都可以用一条命令从一个称为*源*的在线仓库下载安装；为方便全球各地的用户从尽可能近的服务器下载，源通常有很多*镜像*。清华大学开源软件镜像站（https://mirrors.tuna.tsinghua.edu.cn）就是一个国内非常著名的镜像站，它由我校TUNA协会（一个开源爱好者社团）维护，包含了诸多Linux发行版和开源软件的镜像；最重要的是，在校内通过TUNA镜像下载和更新软件不需要流量哦！

Ubuntu使用`apt`包管理器[3]，以下是几个常用命令：

- `apt update`：刷新软件目录（即从源获取所有可以安装的软件的名称和版本等信息）
- `apt upgrade`：升级已经安装的软件到最新版本（注意update和upgrade的区别哦~）

- `apt install`：安装软件
- `apt remove`：卸载软件
- `apt search`：从软件目录中查找软件

上述五个命令中除了`search`以外都需要以Root身份运行。

Tip: 为加快速度和节省流量，强烈建议在自己电脑上更新软件前先切换到TUNA镜像源。方法如下：`sudo vim /etc/apt/sources.list`，使用vim的替换命令：`:%s/cn.archive.ubuntu.com/mirrors.tuna.tsinghua.edu.cn/gg`，其中`cn.archive.ubuntu.com`也有可能是其他域名，反正就是打开文件后看上去满眼都是的那个域名啦😂。敲回车，保存退出。最后`sudo apt update`刷新。

7 IO重定向、管道

练习 编写一个C语言程序，从标准输入读取一个字符串`name`，打印`Hello, ...`，其中省略号表示`name`。用`gcc`编译并运行。

如果我们希望从文件输入输出呢？不需要修改代码，只需使用IO重定向就可以了。

练习 新建一个文件`in.txt`，内容为需要显示的名字。（假设前一个练习得到的可执行文件叫`a.out`）运行以下命令：

- `./a.out < in.txt`
- `./a.out > out.txt`
- `./a.out < in.txt > out.txt`
- `echo Kitty | ./a.out`
- `echo Kitty | ./a.out > out.txt`

运行完第1, 3, 5个命令后，用`cat out.txt`查看`out.txt`的内容。

用`<`, `>`进行重定向应该比较容易理解，而`|`又是什么呢？`|`称为管道符号，作用是把前一个命令的标准输出作为后一个命令的标准输入。这是Unix中最朴素的进程间通信的方法。

重定向应用举例

练习 创建两个文件`1.txt`, `2.txt`，内容随便写一点东西。然后运行`cat 1.txt 2.txt > 3.txt`，查看`3.txt`的内容。

知道`cat`(concatenate)的本意了吧？

管道应用举例

这里介绍一个叫`grep`的命令，它的作用是从标准输入读取文本，如果某一行能够被`grep`的参数所匹配就输出，否则不输出。

不知所云？来看一个例子：

```
➜  /home ls -l
total 88
drwxr-xr-x 10 asta    asta     4096 Jul 11 11:53 asta
drwxr-xr-x  2 chengxt tutorial 4096 Jul 10 00:45 chengxt
drwxr-xr-x  2 guopk   tutorial 4096 Jul 10 00:45 guopk
drwxr-xr-x  5 haocf   tutorial 4096 Jul 11 11:11 haocf
drwxr-xr-x  3 huangr  tutorial 4096 Jul 11 11:00 huangr
drwxr-xr-x  3 huangsc tutorial 4096 Jul 11 11:12 huangsc
drwxr-xr-x  2 kangwx  tutorial 4096 Jul 10 00:45 kangwx
drwxr-xr-x  2 liz     tutorial 4096 Jul 10 00:45 liz
drwxr-xr-x  3 louzq   tutorial 4096 Jul 11 09:46 louzq
drwxr-xr-x  2 luyf    tutorial 4096 Jul 10 00:45 luyf
drwx--x---  8 luyw    asta     4096 Jul 11 12:14 luyw
drwxr-xr-x  2 sunzl   tutorial 4096 Jul 10 00:45 sunzl
drwxr-xr-x  3 tangxw  tutorial 4096 Jul 11 09:47 tangxw
drwxr-xr-x  3 taohz   tutorial 4096 Jul 10 18:30 taohz
drwxr-xr-x  3 tiany   tutorial 4096 Jul 10 00:46 tiany
drwxr-xr-x  2 wangqt  tutorial 4096 Jul 10 00:45 wangqt
drwxr-xr-x  3 wangyh  tutorial 4096 Jul 10 22:39 wangyh
drwxr-xr-x  2 weiyx   tutorial 4096 Jul 10 00:45 weiyx
drwxr-xr-x  3 wut     tutorial 4096 Jul 11 11:11 wut
drwxr-xr-x  2 xiaoh   tutorial 4096 Jul 10 00:45 xiaoh
drwxr-xr-x  4 xiazf   tutorial 4096 Jul 10 20:57 xiazf
drwxr-xr-x  2 zhongqy tutorial 4096 Jul 10 00:45 zhongqy
➜  /home ls -l | grep lu
drwxr-xr-x  2 luyf    tutorial 4096 Jul 10 00:45 luyf
drwx--x---  8 luyw    asta     4096 Jul 11 12:14 luyw
```

我们把`ls -l`的一长串输出通过管道输入给`grep`，实现了文本查找的功能。

再举一个很常用的例子：

```
➜  /home ps -aux | grep apache
root     14657  0.0  3.7 424232 38332 ?        Ss   Jul09   0:03 /usr/sbin/apache2
-k start
www-data 26156  0.0  5.9 506832 60572 ?        S    06:25   0:01 /usr/sbin/apache2
-k start
www-data 26893  0.0  7.2 510820 74124 ?        S    09:42   0:00 /usr/sbin/apache2
-k start
www-data 28479  0.0  3.3 427408 34056 ?        S    11:20   0:00 /usr/sbin/apache2
-k start
www-data 28629  0.0  2.2 425296 23156 ?        S    11:24   0:00 /usr/sbin/apache2
-k start
www-data 28630  0.0  3.3 427668 33568 ?        S    11:24   0:00 /usr/sbin/apache2
-k start
www-data 28632  0.0  4.4 427544 45084 ?        S    11:24   0:00 /usr/sbin/apache2
-k start
www-data 28633  0.0  3.8 503500 38884 ?        S    11:24   0:00 /usr/sbin/apache2
-k start
www-data 28634  0.0  3.2 427408 33496 ?        S    11:24   0:00 /usr/sbin/apache2
-k start
www-data 28643  0.0  4.5 432320 45936 ?        S    11:25   0:00 /usr/sbin/apache2
-k start
www-data 28644  0.0  5.1 506888 52492 ?        S    11:25   0:00 /usr/sbin/apache2
-k start
luyw     29054  0.0  0.0  14224   944 pts/1    S+   12:16   0:00 grep --color
apache
```

其中`ps -aux`的作用是罗列系统中所有的进程。我们通过这条命令筛选出了`apache`进程。

事实上，`grep`的强大之处在于它的参数还可以是正则表达式。感兴趣的同学请自行查阅资料。

综合举例：批量创建用户（可选）

```
while read name; do
    suffix=_asta
    echo $name:$(echo -n $name$suffix | sha1sum | awk '{print
$1}')::1001:Example:/home/$name:/bin/bash | sudo newusers
done < username.txt
```

其中：

- `username.txt`的每一行是一个用户名
- `while`，`do`，`done`是Shell中的循环语句
- `newusers`是新建用户的命令，它从标准输入读取参数，参数之间用冒号分隔
- `sha1sum`是求SHA-1值的命令，待哈希的字符串从标准输入读取，如：

```
➜  ~ echo -n luyw_asta | sha1sum
4695e971ca321e5671e642b6ad7e7a7fd119335a  -
```

- 我们发现`sha1sum`的最后有空格和横杠，我们希望把它们去掉。于是我们就使用了`awk`这款（非常用户不友好）的命令行文本处理神器。（其实关于的`awk`这条命令我也是从网上抄来的啦）

8 用`scp`传输文件

最后补充一下在本地和服务器之间传输文件的方法。

`scp`的用法跟`cp`是非常接近的。比如要把本地一个叫`example.txt`的文件拷到服务器上我的用户的`~`目录下，在本地的Git Bash（或其他shell）中运行：

```
scp example.txt luyw@daasta.cn:~
```

简而言之，远程的路径用`用户名@服务器地址:远程路径`的语法表示就行了，其他都跟`cp`一样。要从远程拷到本地也是一样。

事实上，Windows, Mac和Linux桌面都有工具可以把服务器上的文件系统挂载到本地，那样就可以像把文件从C盘拷到D盘一样轻松地在本地和远程之间互相传输文件了。

9 下一步?

- 是不是觉得命令行用得不太爽？推荐一款神器：Oh-my-zsh（https://github.com/robbyrussell/oh-my-zsh）。它不仅颜值高，还提供了很多有用的快捷命令和插件。你会喜欢上它的~
- 试着在Linux命令行下使用一些常用的编程工具，比如`gcc`, `python`, `git`, `cmake`。其中CMake可以创建类似VS工程的工程，是创建超过一个文件的C/C++工程时首选的跨平台解决方案。
- 有兴趣深入学习Linux命令行的同学可以阅读《鸟哥的Linux私房菜》这套书，我会上传到网盘上。但是它篇幅很大而且讲得比较细，除非对这方面有强烈兴趣或者闲得无聊，否则个人不是很推荐完整阅读。作为参考还是不错的。
- 遇到问题的时候，除了man page外，Google和StackOverflow永远是你的好帮手！

---

1. 严格来说，Linux只包括操作系统内核，而本教程中介绍的大部分工具都由GNU工程提供。我们通常所说的"Linux"其实应该称作"GNU/Linux"。在这里我们暂且不必纠结于术语问题。↩

2. 127.0.0.1是一个特殊的IP地址，表示「本机」。↩

3. 低于16.04的版本略有不同：没有统一的`apt`命令，而分散在`apt-get`、`apt-cache`等多个不同命令中。↩