

# LPI

Terminology:

Diagram نقشه شبکه

Header شناسنامه

Document منابع

Encapsulation کپسوله سازی

Encapsulate کپسوله کردن

Application برنامه

Transport حمل و نقل

Network شبکه

DataLink لینک اطلاعات

Physical فیزیکی

Ventors فروشنده‌گان

Presentation آماده سازی

Session جلسه

Basic مقدماتی

sHost میزبان

Node گره

Consortium کنسرسیوم

Wireless lan شبکه داخلی بی سیم

Configure پیکربندی

Wiggle Room امکان مانور

Site License لایسنس درون سازمانی

GUI رابط گرافیکی کاربر

Shareware اشتراک افزار

Freeware نرم افزار رایگان

Regular Expression عبارت باقاعده

# CHAPTER 1

## Selecting an Operating System

این حقیقت که شما در حال خواندن این کتاب هستید به این معنی است که می خواهید در مورد سیستم عامل لینوکس بیاموزید.

برای شروع این سفر، شما باید ابتدا بدانید لینوکس چیست و سیستم عامل چیست. این فصل توضیح می دهد سیستم عامل چیست، کاربران چطور با سیستم عامل در ارتباط هستند، چگونگی ارتباط لینوکس با دیگر سیستم عامل های معروف و حتی نحوه پیاده سازی لینوکس های خاص چگونه است. فهمیدن این مسائل میتواند به شما کمک کند به لینوکس سوئیچ کنید و در مورد سیستم های مختلف بر پایه لینوکس یاد بگیرید.

### What Is an OS?

سیستم عامل تمام قابلیت های اساسی یک سیستم را فراهم می کند، حداقل از دید یک نرم افزار. سیستم عامل به شما اجازه میدهد از سخت افزار سیستم استفاده کنید، استاندارد های رابط کاربری را تعریف میکند، و ابزار ابتدایی که به برنامه ها اجازه اجرا شدن روی کامپیوتر را میدهد را فراهم میکند. این بخش توضیح میدهد که بخش های مختلفی که سیستم عامل را تشکیل میدهند و چطور کار کردن این ها با هم برای ایجاد حس محاسبه است.

### What Is a Kernel?

کرنل یک سیستم عامل، یک جزء نرم افزاری است که مسئول مدیریت فیچرهای مختلف سطح پایین کامپیوتر است که شامل:

- ایجاد ارتباط با سخت افزارها (آداتوهای شبکه، هارد دیسک ها و ...)
- اختصاص دادن رم به برنامه های مجزا
- اختصاص دادن زمان پردازش به برنامه های مجزا
- فعال کردن برنامه ها برای تعامل با دیگر برنامه ها

وقتی شما از یک برنامه استفاده می کنید (مثلا مرورگر) این برنامه با تکیه بر کرنل بسیاری از عملکردهای اساسی را انجام میدهد. مرورگر میتواند با دنیای بیرون با استفاده از قابلیت های شبکه ای فراهم شده توسط کرنل ارتباط برقرار کند. کرنل به مرورگر، رم و زمان پردازش می دهد که بدون آتها نمی تواند اجرا شود. ممکن

است مرورگر برای نمایش محتوای مالتی مدیا به پلاگین نیاز داشته باشد؛ چنین برنامه‌هایی از طریق سرویس‌های کرنل با مرورگر در تعامل هستند و اجرا می‌شوند. هر برنامه‌ای که اجرا می‌کنید با روش مشابهی به کرنل تکیه می‌کند، اگرچه جزئیات از یک سیستم عامل به دیگری و از یک برنامه به دیگری متفاوت می‌باشد. کرنل نرم افزاری است که مثل چسب کامپیوتر کنارهم نگه می‌دارد. بدون کرنل کامپیوتر مدرن توانایی بسیار کمی دارد.

کرنل‌ها قابل تعویض نیستند، کرنل لینوکسی با کرنل سیستم‌های بسیار قدرتمند و لپ‌تاپ اپل استفاده می‌شود، و همچنین با کرنل ویندوز که در لپ‌تاپ و سیستم‌های قدرتمند سازگار با مایکروسافت استفاده می‌شود متفاوت است.

**نکته:** بعضی برنامه‌ها با کرنل‌های متعدد اجرا می‌شوند، اما اکثراً به ترفندهای اختصاصی سیستم عامل نیاز دارند. برنامه نویس‌ها فایل‌های باینری را ایجاد می‌کنند (فایل‌های برنامه برای پردازنده‌های خاص و کرنل برای هر سیستم عامل است). شما باید فایل باینری ای را اجرا کنید که مختص سیستم عاملی است که برنامه روی آن اجرا می‌شود.

لینوکس از کرنلی استفاده می‌کند به نام لینوکس، در واقع از نظر فنی کلمه لینوکس فقط به کرنل بر می‌گردد. ویژگی دیگری که شما ممکن است در لینوکس با آن مواجه شوید ارائه برنامه‌های بدون کرنل است، که اکثر آنها روی پلتفرم‌های دیگر در دسترس است، همانطور که به صورت خلاصه در "What Else Identifies as OS" توضیح داده شد.

دانش آموزی به نام لینوس توروالدز کرنل لینوکس را در سال ۱۹۹۱ ساخت. لینوکس به طور قابل توجهی از آن زمان تکامل یافته است. امروزه لینوکس بر روی طیف گسترده‌ای از پردازنده‌ها و سخت افزارها قابل اجراست. ساده ترین راه برای یادگیری لینوکس استفاده از آن بر روی سیستم‌های خانگی و لپ‌تاپ هاست و این نوع از پیکربندی است که در این کتاب بر آن تاکید شده است. به هر حال کرنل لینوکس روی همه چیز اجرا می‌شود، از موبایل‌های خیلی کوچک گرفته تا ابر کامپیوتر‌های قدرتمند اجرا می‌شود.

## What Else Identifies an OS?

کرنل هسته مرکزی هر سیستم عامل است اما عنصری است که بیشتر کاربران نمی‌توانند مستقیماً آن را دستکاری کنند. در عوض بسیاری از کاربران با تعدادی از اجزای دیگر نرم افزار در تعامل اند که بسیاری از آنها ارتباط نزدیکی با سیستم عامل‌های خاص دارند. چنین برنامه‌ها شامل موارد زیر است:

**Command-Line Shell** : سال‌ها پیش کاربران تنها از طریق نوشتن دستورها در یک برنامه (که به عنوان *Shell* شناخته می‌شود) که چنین دستوراتی در آن پذیرفته می‌شوند با کامپیوتر ارتباط داشتند. دستورات اسم فایل‌ها را تغییر میدهند، برنامه‌ها را اجرا می‌کنند و... . اگرچه امروزه کاربران از حالت متنی شل‌ها استفاده نمی‌کنند، اما برای کاربران متوسط و حرفه‌ای لینوکس مهم است، پس ما آنها را با جزئیات در فصل "Getting to Know The Command Line" 5 توضیح میدهیم و فصل‌های بعد شدیداً بر توانایی شما در

استفاده از حالت متنی شل تکیه میکند. شل های بسیاری در دسترس است، و اینکه کدام شل ها در "Born Again Shell" دسترس و محبوب است بستگی به نوع سیستم عامل دارد. در لینوکس، شل ای به نام "Born Again Shell" مشهور است.

**رابط گرافیکی کاربر:** رابط گرافیکی کاربر(GUI) یک بهبود در حالت شل است، حداقل برای کاربر مبتدی. رابط گرافیکی کاربر به جای دستور تایپی به آیکون ها، منوها و نشانگر موس تکیه می کند. ویندوزها و سیستم عامل های مک هر دو رابط های گرافیکی خاص خود را دارند. لینوکس به رابط گرافیکی کاربر خود به نام X Window System یا به اختصار X تکیه میکند. X یک GUI خیلی ابتدایی است، پس همچنین لینوکس از برنامه محیط دسکتاپ استفاده می کند، مانند محیط (GNOME) یا (KDE Desktop Environment) برای ارائه تجربه کاربری کامل تر. اختلافات بین محیط کاربری دسکتاپ لینوکس و رابطهای گرافیکی در ویندوز یا مک احتمالاً اولین چیزی است که وقتی که برای اولین بار از لینوکس استفاده میکنید، بیشتر به چشم می آید.

**برنامه های ابزاری:** سیستم عامل های مدرن همواره با طیف گسترده ای از برنامه های ابزاری ساده عرضه می شوند. ماشین حساب، تقویم، ادیتور متن، ابزار نگهداری دیسک و امثال اینها. این برنامه ها بین سیستم عامل ها متفاوت است. حتی اسمی و روش های راه اندازی آنها نیز می تواند بین سیستم عامل ها متفاوت باشد.

خوبی‌ترین شما معمولاً می توانید برنامه های مورد نظر خود را با بررسی منوها در محیط اصلی دسکتاپ پیدا کنید.

**کتابخانه ها:** تا زمانی که برنامه نویس نباشید، بعید است که با کتابخانه ها به طور مستقیم کار کنید. با این حال ما آنها را در لیست قرار می دهیم زیرا آنها خدمات حیاتی ای به برنامه ها ارائه می دهند. کتابخانه ها مجموعه ای از توابع برنامه نویسی است که می تواند توسط برنامه ها مورد استفاده قرار گیرد. برای مثال بیشتر برنامه ها در لینوکس به کتابخانه ای بنام `libc` متکی است. سایر کتابخانه ها ویژگی هایی مرتبط با رابط گرافیکی کاربر ارائه می دهند یا به آنها کمک می کند گزینه های ارسال شده در کامند لاین را تجزیه کنند. کتابخانه های زیادی برای لینوکس وجود دارد که به غنی سازی چشم انداز نرم افزار لینوکس کمک می کند. برنامه های کاربردی: عمدۀ برنامه های کاربردی - موتور های جستجو، پردازشگر متنی، ادیتور گرافیکی و... دلایل معمول برای استفاده از کامپیوتر ها هستند. اگرچه برنامه های این چنینی معمولاً به صورت فنی از سیستم های عامل جدا هستند اما گاهی اوقات با سیستم عامل های خاصی مرتبط میشوند. حتی اگر برنامه ای روی تعداد زیادی از سیستم عامل ها در دسترس باشد ممکن است تفاوت روی هر سیستم عامل احساس شود که دلیل آن تفاوت در رابط گرافیکی کاربر و ویژگی های خاص سیستم عامل است.

**نکته:** می توانید معادل های لینوکسی را برای برنامه های سیستم عامل های محبوب مک یا ویندوز را روی وب سایت های نرم افزارهای اپن سورس مانند [www.linuxalt.com](http://www.linuxalt.com) جستجو کنید.

**نکته:** علاوه بر نرم افزاری که روی سیستم عامل اجرا می شود، چندین ویژگی دیگر نیز وجود دارد که می تواند بین سیستم عامل ها تمایز ایجاد کند مانند جزئیات اکانت های کاربر، قوانین نام گذاری فایل های دیسک و جزئیات فنی راه اندازی سیستم. تمام این امکانات به وسیله نرم افزاری که جزوی از سیستم عامل است

کنترل می شود، که بعضی اوقات توسط کرنل و بعضی اوقات توسط برنامه های غیر کرنلی تحت کنترل می باشد.

## Investigating User Interfaces

قبل‌ا به تمایز بین حالت متنی و رابط گرافیکی کاربر اشاره کردیم. اگرچه اکثر کاربران به دلیل سهولت استفاده از GUI طرفدار آن هستند ولی لینوکس سنت قدیمی و قدرتمند حالت متنی را حفظ کرده است. فصل 5 ابزار حالت متنی را با جزئیات توضیح می دهد و در فصل 4 "Using Common Linux Programs" اصول اولیه عملیات رابط گرافیکی کاربر را پوشش می دهد. این که شما پیش زمینه ای از اصول اساسی در هر دو حالت متنی و رابط گرافیکی کاربر داشته باشید مهم است چرا که در فصل های میانی مشکلات رابط گرافیکی کاربر ظاهر می شوند.

## Using a Text-Mode User Interface

در گذشته و حتی گاهی امروزه کامپیوتر های لینوکس در حالت متنی بوت می شوند. پس از بوت شدن کامل سیستم یک اعلان ورود در حالت ساده نمایش می دهد که ممکن است به این حالت شبیه باشد:  
Fedora 30 (Workstation Edition)  
Kernel 5.0.9-301.fc30.x86\_64 on an x86\_64 (tty1)

نکته: برای امتحان کردن ورود به حالت متنی در ابتدا شما باید لینوکس را روی کامپیوتر نصب کنید. نه امتحان ضروریات لینوکس و نه این کتاب نصب و راه اندازی لینوکس را پوشش نمی دهند. برای یادگیری بیشتر به اسناد توزیع خود مراجعه کنید.

جزئیات پرامپت لاجین به سیستم از سیستم دیگر متفاوت است. این مثال شامل چندین بخش از اطلاعات است:

- نام و نسخه سیستم عامل: 30 Fedora Linux version
- نام کامپیوتر: essentials
- نام سخت افزاری دستگاه که برای ورود استفاده می شود: tty1
- درخواست ورود به سیستم خودش: login:

نکته: اگر پرامپت لاجین به GUI را مشاهده کردید می توانید یک پرامپت متنی را با فشردن Ctrl+Alt+F2 یا Ctrl+Alt+F3 فعال کنید. برای بازگشت به پرامپت لاجین سیستم در GUI Ctrl+Alt+F7 یا Ctrl+Alt+F1 را بزنید.

برای لگین به چنین سیستمی باید نام کاربری را خود را در قسمت `login:` وارد کنید. سپس سیستم از شما یک رمز عبور میخواهد که شما نیز باید آن را بنویسید. اگر شما نام کاربری و رمز عبور را صحیح وارد کنید، احتمالاً کامپیوتر یک پیام ورود به سیستم و به دنبال آن یک پیام شل نمایش می‌دهد:

```
[rich@essentials:~]$
```

در این کتاب، ما اکثر قسمتی از دستورات مثال که در خطوط مجزا ظاهر می‌شوند را حذف می‌کنیم. با این حال، برای دستورات معمولی کاربران، نشان دلار (\$) را حفظ می‌کنیم. برخی از دستورات باید به عنوان حساب کاربری `root` وارد شوند، که حساب کاربری مدیریتی لینوکس است. برای چنین دستوراتی، نشان را به علامت (#) تغییر می‌دهیم، زیرا بیشتر توزیع‌های لینوکس تغییر مشابهی را به نشان‌های خود برای کاربر `root` اعمال می‌کنند.

نکته: فصل 13 "Creating Users and Groups"، حساب‌های لینوکس که شامل حساب کاربری `root` است را با جزئیات توضیح می‌دهد.

جزئیات پaramپت شل از یک نصب به دیگر متغیر است، اما شما می‌توانید دستورات حالت متنی را در پaramپت شل تایپ کنید. به عنوان مثال، می‌توانید `ls` (مخفف list) را تایپ کنید تا لیستی از فایل‌ها در دایرکتوری فعلی را ببینید. پایه‌ترین دستورات با حذف حروف صدادار و گاهی حروف ساکن مخفف شده‌اند تا میزان تایپ مورد نیاز برای اجرای یک دستور حداقل شود. این تاثیر نامطلوبی دارد و باعث می‌شود بسیاری از دستورات نسبتاً مبهم باشند.

بعضی دستورات هیچ اطلاعاتی را نمایش نمی‌دهند، اما بیشتر آنها نوعی خروجی تولید می‌کنند. برای مثال دستور `ls` لیستی از فایل‌ها تولید می‌کند:

```
$ ls  
chapter1.doc figure01.png
```

این مثال دو فایل را در فهرست راهنمای فعلی نمایش می‌دهد: `chapter1.doc` و `figure01.png`. شما می‌توانید از دستورات اضافی برای ایجاد تغییر در این فایل‌ها استفاده کنید، مانند `cp` برای کپی آنها، یا `rm` برای پاک کردن (حذف کردن) آنها. فصل 5 "Getting to Know the Command Line" و فصل 7 "Managing Files" بعضی دستورات را ایجاد تغییر در فایل را توصیف می‌کنند. برخی برنامه‌های حالت متنی اسکرین را به منظور ارائه بروزرسانی‌های مداوم یا قادر ساختن شما به تعامل با داده‌ها به شوه‌ای انعطاف‌پذیر، در اختیار می‌گیرند. برای مثال شکل 1.1، ویرایشگر متنی `nano` را نشان میدهد، که با جزئیات در فصل 10 "Editing Files" توضیح داده شده. وقتی `nano` کار می‌کند شما می‌توانید از جهت‌های صفحه کلید برای حرکت به اطراف استفاده کنید، با تایپ کردن متن اضافه کنید و...

```

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108:/home/syslog:/bin/false
_apt:x:105:65534:/nonexistent:/bin/false
messagebus:x:106:110:/:/var/run/dbus:/bin/false
uuid:x:107:111:/:/run/uuid:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
ntp:x:109:116:/:/home/ntp:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:111:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/bin/false
colord:x:113:123:colord colour management daemon,,,:/var/lib/colord:/bin/false
geoclue:x:114:124:/:/var/lib/geoclue:/bin/false
speech-dispatcher:x:115:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
hplip:x:116:7:HPLIP system,,,:/var/run/hplip:/bin/false
kernooops:x:117:65534:Kernel Oops Tracking Daemon,,,:/bin/false
pulse:x:118:125:PulseAudio daemon,,,:/var/run/pulse:/bin/false

```

[ Read 43 lines (Warning: No write permission) ]

Get Help   Write Out   Where Is   Cut Text   Justify   Cur Pos   Prev Page   First Line   WhereIs Next  
Exit   Read File   Replace   Uncut Text   To Spell   Go To Line   Next Page   Last Line   To Bracket

Figure 1.1 Some text-mode programs take over the entire display.

حتی اگر از لاگین گرافیکی استفاده می کنید، می توانید از شل حالت متنی در داخل پنجره استفاده کنید که به عنوان ترمینال شناخته می شود. با GUI رایج لینوکس می توانید یک برنامه ترمینال اجرا کنید که یک پرامپت شل و ابزارهایی برای اجرای حالت متنی برنامه فراهم می کند.

## Using a Graphical User Interface

اکثر کاربران با GUI نسبت به حالت متنی راحت ترند. بنابراین خیلی از سیستم های مدرن لینوکس به طور پیش فرض با GUI مشابه آنچه در شکل 1.2 است شروع می شوند و بر روی صفحه لاجین ارائه می شود. شما می توانید نام کاربری را از لیست انتخاب یا آن را بنویسید و در ادامه با نوشتن رمز عبور لاجین کنید.

**نکته:** بعضی از صفحه های ورود رابط گرافیکی کاربر لینوکس تا زمانی که یک نام کاربری معتبر وارد کنید شما را برای رمز عبور اجبار نمی کنند.

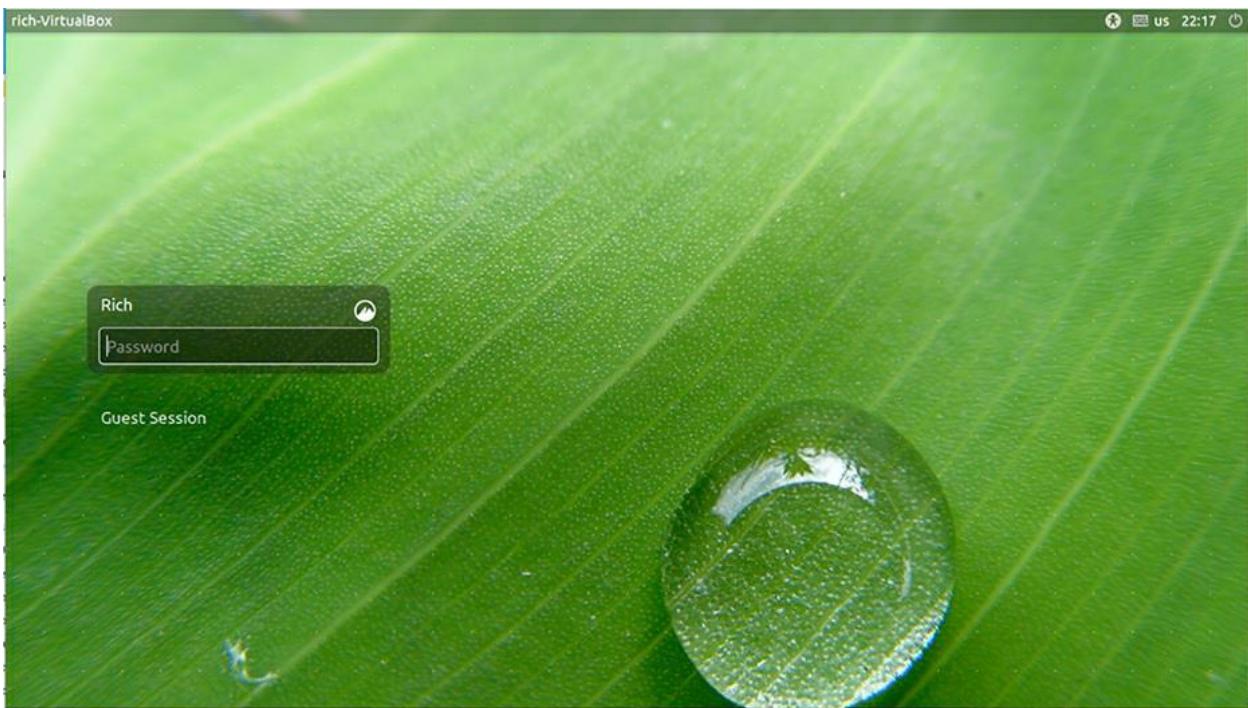


Figure 1.2 Graphical login screens on Linux are similar to those for Windows or macOS.

برخلاف ویندوزها و مک ها، لینوکس تعدادی محیط دسکتاپی مختلف برای شما فراهم میکند که از بین آنها انتخاب کنید. اینکه از کدامیک استفاده می کنید بستگی به نوع لینوکس مورد استفاده تان، گزینه های نرم افزاری انتخاب شده در زمان نصب و ترجیحات شخصی خودتان دارد. انتخاب های رایج شامل GNOME, KDE Plasma, Cinnamon, and Xfce میباشد. بسیاری از گزینه های دیگر نیز در دسترس هستند. بسیاری از دسکتاپ های گرافیکی به طور پیش فرض دارای امکانات فناوری کمکی هستند. در شکل 1.2 آیکون آدمک در گوش سمت راست بالای پنجره ورود به سیستم فدورا به شما این امکان را می دهد که یک فناوری کمکی مانند Screen Reader یا کیبورد روی صفحه، برای کمک به لاگین به سیستم انتخاب کنید. محیط های دسکتاپ لینوکس می توانند کاملا از یکدیگر متفاوت به نظر برسند، اما همه می آنها عملکرد مشابهی را ارائه می دهند. شکل 1.3 حالت پیش فرض دسکتاپ Cinnamon بر روی توزیع Mint 18.3 با چند برنامه در حال اجرا را نشان می دهد. فصل 4 محیط های معمول دسکتاپ و ویژگی های آنها را با جزئیات بیشتری توضیح می دهد، اما در حال حاضر، باید بدانید که همه آنها ویژگی هایی مانند موارد زیر را ارائه می دهند:

لانچر برنامه ها: شما میتوانید برنامه ها را از فهرست ها یا لیست ها انتخاب کنید. معمولا یک یا چند فهرست در بالا، پایین یا کنار صفحه قرار دارند. در شکل 1.3 شما می توانید بر روی آیکون برگ نعناع در گوش می پایین سمت چپ کلیک کنید تا منوی که به آن شکل ظاهر میشود باز کنید.

مدیریت فایل ها: لینوکس GUI مدیریت فایل ها، شبیه به ویندوز ها و سیستم های عامل مک ارائه می دهد. پنجره ای برای یکی از اینها در مرکز شکل 1.3 باز است.

**کنترل پنجره ها:** شما می توانید پنجره ها را با کلیک و جابه جا کردن نوار عنوان آنها تکان دهید، سایز آن را با کلیک کردن و جابه جا کردن گوشه های آن تغییر دهید و ...

**دسکتاپ های چندتایی:** اکثر محیط های دسکتاپ لینوکس به شما قابلیت این را می دهد که چند دسکتاپ مجازی فعال داشته باشید که هر کدام با برنامه های خودش تنظیم شده است. این قابلیت به شما کمک می کند در زمانی که شما چندین برنامه به طور همزمان اجرا می کنید صفحه تان مرتب شده باشد. معمولاً یک آیکون از منو به شما قابلیت جابجایی بین دسکتاپ های مجازی را می دهد.

**گزینه های خروج:** شما می توانید از اکانت لینوکس خود خارج شوید که به شما این قابلیت را می دهد که کامپیوتر خود را خاموش کنید یا به کاربر دیگری اجازه ورود دهید.

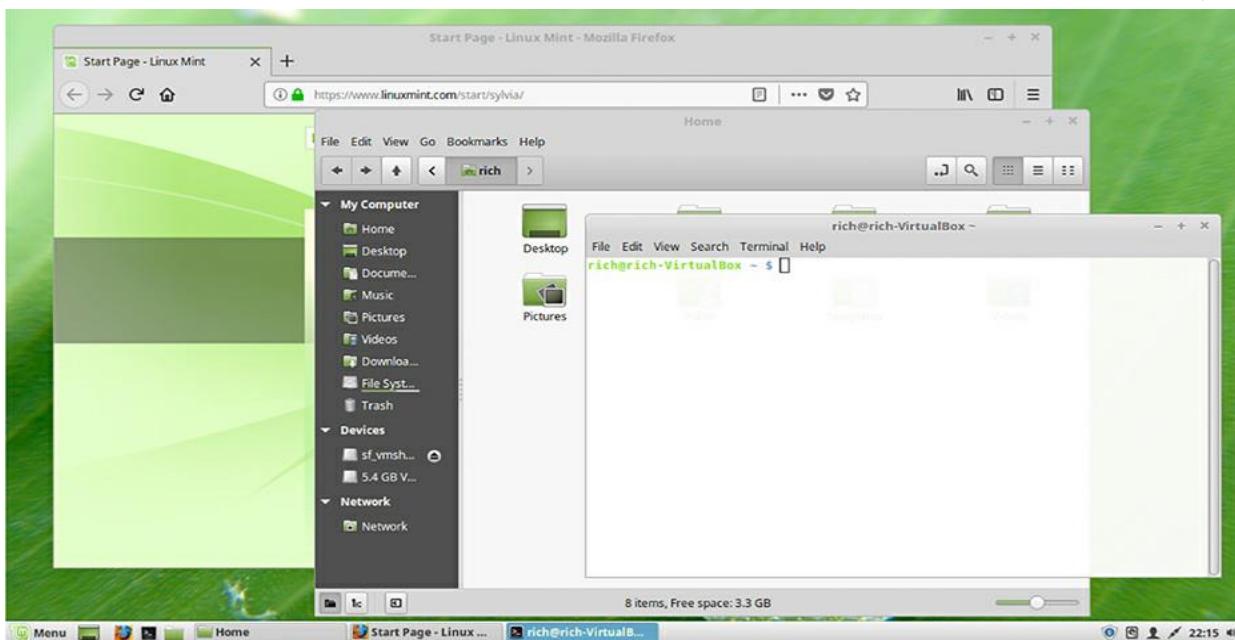


Figure 1.3 Linux desktop environments provide the types of GUI controls that most users expect.

**هشدار:** خروج از حساب کاربری در محیط های عمومی بسیار مهم است. اگر شما از حساب کاربری خود خارج نشوید، ممکن است یک غریبه از حساب کاربری شما برای اهداف خبیثانه استفاده کند.

هر چه بیشتر با لینوکس آشنا می شوید، متوجه خواهد شد که محیط های گرافیکی (GUI) آن بسیار انعطاف پذیر هستند. اگر متوجه شوید که محیط پیشفرض توزیع شما برایتان جذاب نیست، می توانید آن را تغییر دهید. هر چند که تمامی آنها ویژگی های مشابهی را ارائه می دهند، اما برخی از افراد ترجیحات قوی تری نسبت به محیط های دسکتاپ دارند. لینوکس به شما این امکان را می دهد که انتخاب کنید، که این امکان در ویندوز یا مک اواس موجود نیست. بنابراین احساس راحتی کنید و محیط های دسکتاپ مختلف را امتحان کنید.

نکته: شاید نیاز باشد که محیطهای دسکتاپ اضافی را نصب کنید تا از آنها استفاده کنید. این موضوع در این کتاب پوشش داده نشده است.

## Where Does Linux Fit in the OS World?

عنوان این فصل اشاره به یک مقایسه دارد، و از آنجا که این کتاب در مورد لینوکس است، مقایسه باید با سیستم عامل یا خانواده سیستم عامل دیگر صورت گیرد: یونیکس، اپل macOS و مایکروسافت ویندوز.

نکته: همانطور که بعدا در "What Is a Distribution?" توضیح میدهیم، لینوکس را می‌توان خانواده ای از سیستم عامل‌ها در نظر گرفت.

## Comparing Linux to Unix

اگر بخواهید یک "شجره خانواده" از سیستم عامل‌ها ترسیم کنید، نیاز به وقت زیادی خواهد داشت. این به این دلیل است که طراحان سیستم عامل اغلب ویژگی‌های یکدیگر را تقلید و گاهی اوقات حتی کد یکدیگر را در عملکرد سیستم عامل خود وارد می‌کنند. نتیجه می‌تواند یک آشفتگی از شباهت‌های درهم بین سیستم عامل‌ها باشد که دلایل آن از تصادف تا "فرض گرفتن" کد را شامل می‌شود. تلاش برای ترسیم این تاثیرات می‌تواند بسیار دشوار باشد. اما در مورد لینوکس و یونیکس، یک عبارت کلی درست است: لینوکس از روی یونیکس طراحی شده است.

یونیکس در سال 1969 در آزمایشگاه بی‌AT&T ایجاد شد. یونیکس تاریخچه‌ای پیچیده دارد و شامل چندین شاخه (یعنی تقسیم کد به دو یا چند پروژه مستقل) و حتی بازنویسی کدهای کاملاً مجزا می‌باشد. سیستم‌های لینوکس مدرن، به طور کلی، محصلوی پروژه‌های اپن سورس هستند که برنامه‌های یونیکس را شبیه سازی می‌کنند، یا به طور کلی پروژه‌های اپن سورس اصلی برای نشأت می‌گیرند.

نکته: نرم افزار اپن سورس نرم افزاری است که شما نه تنها می‌توانید آن را اجرا کنید، بلکه خودتان می‌توانید آن را اصلاح و توزیع کنید. فصل 3 "Investigating Linux's Principles and Philosophy"، فلسفه و مسائل حقوقی مربوط به نرم افزار اپن سورس را پوشش میدهد.

این پروژه‌ها شامل:

**کرنل لینوکس:** لینوس توروالدز، کرنل لینوکس را به عنوان یک پروژه برنامه نویسی سرگرمی در سال 1991 ایجاد کرد، اما خیلی زود، بسیار بیشتر از یک سرگرمی شد. کرنل لینوکس به گونه‌ای طراحی شده است که با دیگر کرنل‌های یونیکس سازگار باشد به این معنا که از همان رابط‌های نرم افزاری در سورس کد استفاده می‌کند. این امر استفاده از برنامه‌های اپن سورس برای سایر نسخه‌های یونیکس با کرنل لینوکس را آسان می‌کند.

**پروژه گنو:** پروژه غیر یونیکسی گنو (GNU) تلاشی توسط بنیاد نرم افزار آزاد (FSF) برای توسعه جایگزین‌های اپن سورس برای تمام عناصر اصلی سیستم عامل یونیکس می‌باشد. در سال 1991 FSF مهم‌ترین ابزارهای

این چنینی را به استثنای کرنل منتشر کرده بود. (کرنل گنو HURD اکنون در دسترس است اما به اندازه لینوکس محبوب نیست). ابزارهای جایگزین برای ابزارهای گنو شامل ابزارهای تجاری اختصاصی و ابزارهای اپن سورس توسعه یافته برای انواع BSD Unix است. ابزارهای مورد استفاده در سیستم عامل های شبه یونیکس می توانند بر "طعم" (استعاره از کنو های مختلف) کلی آن تاثیر بگذارند، اما همه این مجموعه ابزارها به اندازه ای مشابه هستند که به هر یونیکس احساس مشابه در مقایسه با سیستم عامل غیر یونیکس می دهند.

نکته: گنو یک نمونه از یک اختصار بازگشته است - یک اختصار که گسترش خود شامل همان اختصار است. این یک مثال از شوخی گیکی است.

**Xorg-X11**: سیستم پنجره Window System X محیط گرافیکی برای بیشتر سیستم عامل های یونیکس است. بیشتر توزیع های لینوکس امروزی از نسخه X 11 از X استفاده می کنند. همانند ابزارهای متنی اصلی ارائه شده توسط پروژه گنو، انتخاب یک سرور X می تواند بر برخی از ویژگی های یک سیستم عامل یونیکس مانند انواع فونت های پشتیبانی شده تأثیر بگذارد. ویلند (Wayland) یک پکیج نرم افزاری جدیدتر از سیستم پنجره X است که در لینوکس استفاده می شود و در حال جلب محبوبیت است.

**محیط های دسکتاپ**: محیط های دسکتاپ اپن سورس مانند GNOME، KDE Plasma، Cinnamon، Xfce و دیگر محیط های محبوب، حتی در نسخه های تجاری یونیکس، به طور گسترده ای از محیط های دسکتاپ تجاری پیشی گرفته اند. بنابراین، در این زمینه شما تفاوت های بزرگی بین لینوکس و یونیکس پیدا نخواهید کرد.

نکته: سیستم عامل مک، که به طور مختصر توضیح داده شد، از لحاظ فنی یک یونیکس تجاری است اما بجای یک محیط دسکتاپ اپن سورس که بر روی Window System X اجرا می شود، از یک محیط گرافیکی کاربری پرچمدار شرکت اپل به نام "Aqua" استفاده می کند.

**برنامه های سرور**: از نظر تاریخی، یونیکس و لینوکس به عنوان سیستم عامل های سرور محبوب بوده اند. سازمان ها از آنها برای اجرای وب سرورها، سرور های پایگاه داده و ... استفاده می کردند. لینوکس همان برنامه های سرور محبوب را اجرا می کند که نسخه های تجاری یونیکس و BSD های اپن سورس اجرا می کنند.

**برنامه های کاربردی کاربری**: در این حوزه، مانند برنامه های سرور، لینوکس همان نرم افزار را اجرا می کند که سایر سیستم عامل های شبه یونیکس را اجرا می کنند. در موارد معده دی، لینوکس برنامه های بیشتری را اجرا می کند، یا بهتر اجرا می کند. این بیشتر به دلیل محبوبیت لینوکس و مجموعه گسترده ای از درایورهای سخت افزاری است که لینوکس ارائه می دهد. برای مثال، اگر برنامه ای نیاز به پشتیبانی کارت گرافیک شبکه داشته باشد، به احتمال زیاد این پشتیبانی را در لینوکس پیدا می کند تا در سیستم عامل های کمتر محبوب یونیکس مانند.

در کل می توان لینوکس را عضوی از خانواده سیستم عامل های شبیه یونیکس دانست. اگرچه لینوکس از نظر فنی یک سیستم عامل یونیکس نیست، اما به اندازه ای مشابه است که تفاوت ها در مقایسه با تفاوت های این خانواده به عنوان یک کل، و سایر سیستم عامل ها، مانند ویندوز، بی اهمیت است.

## Comparing Linux to macOS

سیستم عامل مک اپل یک سیستم عامل تجاری بر پایه یونیکس است که به شدت از BSD ها وام گرفته و رابط کاربری گرافیکی یونیکس (به طور مثال X) را به نفع رابط کاربری خود کنار می‌گذارد. این باعث می‌شود که سیستم عامل مک همزمان بسیار مشابه به لینوکس و بسی متفاوت با آن باشد.

شما می‌توانید یک پنجره ترمینال در سیستم عامل مک باز کنید و خیلی از دستورات توصیف شده در این کتاب رو وارد کنید و به نتایج مشابهی برسید. اگر یک دستور توصیف شده در این کتاب موجود نیست، شما می‌توانید به نحوی آن را نصب کنید. سیستم عامل مک با برخی برنامه‌های محبوب سرور یونیکس عرضه می‌شود، تا شما بتوانید آن را به طوری پیکربندی کنید که مانند لینوکس یا دیگر سیستم عامل‌های بر پایه یونیکس به عنوان یک کامپیوتر سرور شبکه کار کند.

اگرچه سیستم عامل مک در بوزر اینترفیس خود با لینوکس تفاوت دارد. بوزر اینترفیس مک از دیدگاه برنامه نویسی به عنوان *Cocoa* شناخته می‌شود یا به عنوان *Aqua* از دیدگاه کاربر. عناصری که شامل آن می‌شوند به سختی به X و محیط دسکتاپ در لینوکس مشابه است. چون *Cocoa* از لحاظ برنامه نویسی با X سازگار نیست، و برنامه‌ها در سیستم عامل مک نمی‌توانند مستقیماً روی لینوکس اجرا شوند (یا هر سیستم عامل مشابه یونیکس)، و پورت کردن آنها (تغییر دادن سورس کد و کامپایل دوباره) برای لینوکس امر دشواری است. بنابراین، برنامه‌های مک به ندرت به لینوکس منتقل می‌شوند.

مک شامل ابزارهایی از X می‌باشد که تحت *Aqua* اجرا می‌شوند. این باعث می‌شود انتقال ال‌ال‌ال لینوکس و برنامه‌های یونیکس به مک نسبتاً مستقیم و صریح باشد. اگرچه برنامه‌های ناشی از این امر تماماً با بوزر اینترفیس *Aqua* مطابقت ندارند. آنها ممکن است دکمه‌ها، فهرست‌ها و دیگر ویژگی‌هایی داشته باشند که نسبت به ظاهر عادی معادل هایشان در مک بی‌جا و بی مورد بنظر بیایند.

اپل سیستم عامل مک را فقط برای کامپیوتر‌های خودش فراهم می‌کند. ضوابط مجوز آن هرگونه نصب بر روی سخت افزاری که اپل نباشد را منع می‌کند، که البته جدا از مساله مجوزش، نصب سیستم عامل مک روی سخت افزار غیر اپل بسیار دشوار است. یکی از گونه‌های سیستم عامل مک که با نام iOS شناخته می‌شود، روی آی‌پد های اپل و دستگاه و تجهیزات آیفون اجرا می‌شود، و به همان اندازه غیرقابل انتقال به دستگاه‌های دیگر است. بنابراین، سیستم عامل مک عمدها به سخت افزارهای اپل منحصر است. لینوکس در مقابل، بر روی تنوع وسیعی از سخت افزارها، از جمله اکثر PC‌ها اجرا می‌شود و شما حتی می‌توانید لینوکس را روی کامپیوتر‌های مکینتاش نیز نصب کنید.

## Comparing Linux to Windows

اکثر کامپیوترها و لپ‌تاپ‌های امروزی با ویندوز کار می‌کنند. بنابراین، اگر شما لینوکس رو در نظر دارید، نزدیکترین مقایسه با ویندوز است. به طور کلی، لینوکس و

ویندوز امکانات مشابهی دارند؛ هر چند تفاوت های قابل توجهی در جزئیات وجود دارد که شامل موارد زیر میشود:

**مجوزها:** لینوکس یک سیستم عامل اپن سورس میباشد درحالیکه ویندوز یک سیستم عامل انحصاری تجاری است. در بخش ۲ "درک مجوز نرم افزار" به مسائل اپن سورس با جزئیات بیشتری پرداخته میشود، ولی فعلا بهتره بدانید که نرم افزار اپن سورس به نسبت سیستم عامل های انحصاری کنترل بیشتری روی کامپیوتر شما فراهم می کند-البته روی کاغذ. در عمل، شما به تخصص سطح بالایی نیاز دارید تا بتوانید از مزیت های اپن سورس نفع ببرید. اگر شما برای سازمانی که با خرید سنتی نرم افزار راحت تر است کار میکنید در این صورت نرم افزار انحصاری قابل ترجیح تر است. (اگرچه برخی از انواع لینوکس به شکل مشابهی همراه با قرارداد خدمات فروخته میشوند).

**هزینه ها:** خیلی از لینوکس ها بصورت رایگان در دسترس هستند و اگر سعی بر کم کردن هزینه ها دارید بسیار خوشایند هستند. هرچند تخصص لازم برای نصب و نگهداری لینوکس احتمالا بیشتر است و در نتیجه گران تر از تخصص لازم برای نصب و نگهداری ویندوز. مطالعات متفاوتی در رابطه با موضوع هزینه های کلی نگهداری لینوکس و ویندوز انجام شده که نتایج متنوعی داشتند اما اکثرا به نفع لینوکس بوده است.

**سازگاری سخت افزار:** اکثر سخت افزارها معمولاً نیازمند پشتیبانی سیستم عامل در قالب درایور هستند. اکثر تولید کنندگان درایور های ویندوز رو برای دستگاه های خود فراهم میکنند یا با مایکروسافت برای اطمینان از اینکه ویندوز درایور های مناسب را شامل میشود همکاری میکنند. اگرچه برخی تولید کنندگان درایور های لینوکس هم فراهم میکنند، برای اکثر موقع جامعه لینوکس در مجموع باید درایور های لازم رو تامین کنند. این به این معنا است که درایور های لینوکس ممکن است چند هفته و یا حتی ماه ها پس از عرضه شدن تجهیزات حاضر شوند. هم‌زمان، توسعه دهندگان لینوکس مایل هستند تا درایورهای سخت افزارهای قدیمی را برای مدت زمان بیشتری به نسبت خود تولید کنند. بنابراین، یک لینوکس مدرن احتمالا بهتر از ورژن جدید ویندوز روی سخت افزارهای قدیمی اجرا میشود. همچنین لینوکس معمولاً کمتر روی منابع متمرکز میباشد تا عملکرد شما با سخت افزارهای قدیمی کارامد تر باشد.

**دسترسی نرم افزار:** برخی نرم افزارهای محبوب دسکتابپ مانند مایکروسافت آفیس برای ویندوز در دسترس هستند ولی نه برای لینوکس. اگرچه جایگزین های لینوکسی مانند LibreOffice وجود دارند ولی در ذهن مردم جا نیفتاده اند. در حوزه های دیگر، وضعیت بر عکس میباشد. نرم افزار های محبوب سرور، مانند Apache Web Server، در ابتداء برای لینوکس یا یونیکس ایجاد شدند. اگرچه چنین سرورهایی برای ویندوز نیز قابل دسترس هستند اما در لینوکس بسیار کارامد تر اجرا میشوند. اگر شما نرم افزار بخصوصی برای اجرا در نظر دارید، بهتر است اول راجع به دسترسی و عملکرد آن بر روی پلتفرم مدنظر خود تحقیق کنید.

**یوزر اینترفیس:** مانند سیستم عامل مک، ویندوز نیز از یوزر اینترفیس منحصر به فرد خود استفاده میکند. این حقیقت به موضوع ضعف انتقال پذیری بین سیستم عامل ها برمیگردد. (هرچند ابزار هایی برای اتصال بین شکاف سیستم عامل ها وجود دارد؛ ابزار پنجره X برای ویندوز در دسترس میباشدند مثل ابزارهایی که برای اجرای نرم افزارهای ویندوز در لینوکس در دسترس است). برخی کاربرها یوزر اینترفیس ویندوز را به هر فضای دسکتابپ لینوکسی ترجیح میدهند اما برخی دیگر فضای لینوکسی را ترجیح میدهند.

نکته: مایکروسافت در ویندوز ۸ یوزر اینترفیس جدید به نام Metro را معرفی کرد. ایده پشت مترو این است که بر روی هر چیزی از گوشی هوشمند گرفته تا کامپیوتراها اجرا شود. هرچند یوزر اینترفیس مترو خیلی سریع به دلیل دشوار سازی آزادی حرکت برای کاربران با تجربه ویندوز ناپسند شد. بنابراین از یوزر اینترفیس دیفالت ویندوز ۱۰ حذف شد ولی همچنان برای کسانی که آن را ترجیح میدهند در دسترس است.

**قابلیت پیکربندی:** لینوکس به نسبت ویندوز قابلیت پیکربندی بسیار بیشتری دارد. اگرچه هر دو سیستم عامل شرایط لازم برای اجرای نرم افزارهای بخصوص، تغییر تم یوزر اینترفیس و غیره را در زمان راه اندازی فراهم میکنند. ذات اپن سورس بودن لینوکس به این معناست که شما میتوانید هر جزئیاتی که میخواهید را دستکاری کنید. علاوه بر این، میتوانید هر نوع از انواع لینوکس را انتخاب کنید تا شروعی بر تنظیم سیستم به شکلی که خودتان صلاح میدانید داشته باشید.

**امنیت:** طرفداران هر سیستم عامل ادعا میکنند که از دیگری امن تر میباشند. به این دلیل که آنها بر روی موضوعات امنیتی متفاوتی تمرکز میکنند. اکثر تهدیدات برای ویندوز از طرف ویروس ها می آید، که بطور گسترده ویندوز و بستر عظیم کاربرهایش را مورد هدف قرار میدهد. ویروس ها اصولا مشکلی برای لینوکس نیستند؛ در لینوکس، تهدیدات امنیتی اکثرا از سمت نفوذها شامل کاربران غیر قابل اعتماد داخلی و اشکالات پیکربندی سرور می آید.

برای بیش از یک دهه، ویندوز بر عرصه دسکتاپ تسلط داشته است. هم در خانه ها و هم در ادارات، کاربران با ویندوز آشنا شدند و به نرم افزار های محبوب ویندوز عادت کردند، مانند مایکروسافت آفیس. با اینکه لینوکس میتواند در چنین محیط هایی استفاده شود اما از محبوبیت کمتری برخوردار است به دلایل مختلفی مانند ناآشنایی کاربران و این موضوع که ویندوز بر روی خیلی از کامپیوتر ها و لپتاپ ها به صورت از قبل نصب شده عرضه میشود و همچنین کمبود نرم افزارهای لینوکسی که برای اکثر کاربران گیرا باشد.

از طرفی، عموما یونیکس و مخصوصا لینوکس برای فتح بازار سرور آمدند. لینوکس به سرورهای وب، سرورهای دیتابیس، سرورهای ایمیل و غیره که اینترنت را میسازند و بسیاری از کسب و کارها به آنها برای فراهم سازی شبکه های داخلی تکیه میکنند قدرت می بخشد. بنابراین، اکثر کاربران روزانه از لینوکس بدون آنکه متوجه بشوند استفاده میکنند.

در بیشتر موارد، این امکان وجود دارد که یکی از سیستم عامل های لینوکس و ویندوز را روی یک کامپیوتر داشته باشید و عملکرد قابل قبولی را شاهد باشید. اگرچه بعضی اوقات، نیازهای بخصوصی حکم میکند که از چه سیستم عاملی استفاده کنید. برای مثال، شما ممکن است نیاز به اجرای نرم افزار خاصی را داشته باشید، یا ممکن است سخت افزار شما برای ویندوز مدرن قدیمی یا بیش از حد جدید برای لینوکس باشد. در موارد دیگر آشنایی شما یا کاربرهای شما با یک سیستم عامل میتواند ملاک استفاده باشد.

## What Is a Distribution?

تا به الان، ما لینوکس را به گونه ای توصیف کردیم که انگار یک سیستم عامل واحد است اما موضوع به این صورت نمی باشد. تعداد زیادی از توزیع های مختلفی از لینوکس وجود دارد که هرکدام دارای یک کرنل

لینوکس به همراه دسته ای از ابزارها و فایل های پیکربندی میباشد. که باعث میشود دو توزیع لینوکس به همان اندازه ای که با سیستم عامل مک یا ویندوز تفاوت دارند با یکدیگر نیز متفاوت باشد. در نتیجه ما با جزئیات توضیح میدهیم که توزیع چیست، چه توزیع هایی محبوب اند و نگهدارندگان با چه روشی توزیع های خود را بروز نگه میدارند.

## Creating a Complete Linux-Based OS

قبلًا بعضی از اجزایی که یک سیستم عامل لینوکس را میسازد را توضیح دادیم ولی برخی از جزئیات نیازمند تکرار یا توضیح بیشتر هستند:

**یک کرنل(هسته) لینوکس:** مسلماً، کرنل لینوکس در هسته هر سیستم عامل لینوکسی است. ما این عنوان رو با نام یک کرنل لینوکس نوشته ایم به دلیل اینکه کرنل لینوکس دائماً در حال تکامل است. محتمل است که دو توزیع متفاوت از کرنل های کمی متفاوتی استفاده کنند. نگهدارندگان توزیع ها اغلب اوقات با Patch کردن کرنل ها تغییرات کوچک برای رفع باگ ها یا اضافه کردن امکانات جدید ایجاد میکنند.

**ابزار های هسته یونیکس:** ابزارهایی مثل دسته ابزارهای GNU، سیستم پنجره X و ابزارهای مورد استفاده در مدیریت دیسک برای عملکرد عادی یک سیستم لینوکس حیاتی هستند. اکثر توزیع های لینوکس کم و بیش شامل دسته ابزارهای یکسان هستند اما در هسته میتوانند در نسخه ها و Patch های مختلف متنوع باشند.

**نرم افزار تکمیلی:** نرم افزارهای اضافی مثل برنامه های مهم سرور، محیط های دسکتاپ و ابزارهای کارآمد به همراه اکثر توزیع های لینوکس عرضه میشوند و برای نرم افزارهای هسته یونیکس، اکثر توزیع ها گزینه های مشابهی برای چنین نرم افزارهایی فراهم میکنند. اگرچه برخی اوقات توزیع ها "برند" خودشون را مخصوصاً در محیط گرافیکی دسکتاپ فراهم میکنند.

**اسکریپت راه اندازی:** بیشتر "شخصیت" یک توزیع لینوکس از روشنی که پروسه استارت آپ را مدیریت میکند می آید. لینوکس از ابزارها و اسکریپت ها برای اجرا کردن ده ها برنامه که کامپیوتر را به یک شبکه متصل میکند برای مهیا کردن صفحه لاگین و غیره استفاده میکند.

**نصب کننده:** نرم افزار برای استفاده شدن باید نصب شود و اکثر توزیع های لینوکس نرم افزار نصب خاصی برای کمک به شما در مدیریت کارهای مهم فراهم میکند. در نتیجه دو توزیع مختلف ممکن است پروسه نصب متفاوتی داشته باشند که به شما اختیاراتی در امکانات کلیدی مثل چینش های دیسک و ساخت اولیه اکانت کاربر میدهد.

معمولًا توزیع های لینوکس از طریق وبسایت خود برای دانلود قابل دسترس هستند. شما معمولاً میتوانید یک CD-R یا DVD Image File از توزیع مورد نظر را دانلود کنید و به روی یک دیسک نوری انتقال دهید. زمانی که دیسک بدست آمده را راه اندازی می کنید یک نصب کننده اجرا میشود و شما میتوانید سیستم عامل را نصب کنید. اگر کامپیوتر شما قادر درایو دیسک نوری میباشد شما همچنین میتوانید از همان فایل برای ساخت یک USB قابل اجرا استفاده کنید همچنین نسخه های ابری بسیاری از توزیع های لینوکس وجود دارد که به شما این اختیار را میدهد که یک سیستم لینوکس کامل را دانلود کنید و بر روی یک ماشین مجازی و یا

بر روی یک سرویس ابری تجاری مثل سرویس وب آمازون (AWS) یا پلتفرم ابری گوگل (google cloud) اجرا کنید.

**پیشنهاد:** اگر برای امتحان کردن لینوکس کنجکاوید و لپ تاپ یا سیستم دیگری در دسترس ندارید میتوانید از نرم افزار های ماشین مجازی مثل VirtualBox یا VMWare استفاده کنین که به شما این اجازه را میدهد که در داخل یک ویندوز یا مک، لینوکس را اجرا کنید بدون اینکه سیستم عامل فعلی خود را تغییر دهید.

برخی نصب کننده های لینوکس با تمامی نرم افزارهایی که شما احتمالا نیاز داشته باشید عرضه میشود. باقی توزیع ها با حداقل نرم افزار عرضه میشوند و انتظار میروند که شما اتصال فعالی به اینترنت داشته باشید تا نصب کننده بتواند نرم افزارهای اضافی را دانلود کند. اگر کامپیوتر شما به اینترنت متصل نمی باشد مطمئن شوید که نصب کننده مناسبی را تهیه می کنید.

## A Summary of Common Linux Distributions

بسته به نحوه شمارش، حدود دوازده توزیع اصلی لینوکس برای کامپیوترهای رومیزی، لپتاپ، و کامپیوترهای سرور کوچک و صدها توزیع دیگر وجود دارد که اهداف تخصصی را دنبال میکنند. جدول 1.1 ویژگی های مهم ترین توزیع ها را خلاصه می کند.

Table 1.1 Features of major Linux distributions

Distribution	Availability	Package Format	Release Cycle	Administrator skill requirements
Arch	Free	Pacman	Rolling	Expert
CentOS	Free	RPM	approximately 2-year	Intermediate
Debian	Free	Debian	2-year	Intermediate to expert
Fedora	Free	RPM	approximately 6-month	Intermediate
Gentoo	Free	Ebuild	Rolling	Expert

Mint	Free	Debian	6-month	Novice to intermediate
openSUSE	Free	RPM	8-month	Intermediate
Red Hat Enterprise	Commercial	RPM	approximately 2-year	Intermediate
Slackware	Free	Tarballs	Irregular	Expert
SUSE Enterprise	Commercial	RPM	2–3 years	Intermediate
Ubuntu	Free	Debian	6-month	Novice to intermediate

این ویژگی ها نیاز به توضیح دارند (از چپ به راست):

**Availability:** اکثر توزیع های لینوکس کاملاً اپن سورس یا نرم افزار رایگان هستند. با این حال، برخی شامل اجزای اختصاصی هستند و معمولاً با یک قرارداد پشتیبانی به قیمت پول فروخته می شوند. (Red Hat SUSE Enterprise Linux و Enterprise Linux (RHEL) دو نمونه برجسته از این نوع توزیع هستند. هر دو پسر عمومی کاملاً رایگان دارند. برای RHEL، CentOS یک کلون نزدیک است که اجزای اختصاصی را حذف می کند، و فدورا یک نسخه باز است که به عنوان یک بستر آزمایشی برای فناوری هایی عمل می کند که ممکن است در نهایت در RHEL گنجانده شوند. برای SUSE Enterprise، openSUSE یک جایگزین رایگان است.

**Package Format:** اکثر توزیع های لینوکس نرم افزار را در بسته هایی توزیع می کنند که مجموعه ای از بسیاری از فایل ها در یک بسته هستند. نرم افزار Package یک پایگاه داده محلی از فایل های نصب شده را نگهداری می کند و ارتقاء و حذف نصب را آسان می کند. سیستم مدیریت بسته (RPM) محبوب ترین سیستم در دنیای لینوکس است، اما بسته های دبیان نیز بسیار رایج هستند. سایر سیستم های بسته بندی خوب کار می کنند، اما مختص توزیع هستند، مانند سیستم مدیریت بسته های pacman که در آرچ لینوکس استفاده می شود. Slackware از این جهت غیر معمول است که از tarball برای بسته های خود استفاده می کند. اینها فایل های بسته ای هستند که توسط ابزار استاندارد tar ایجاد شده اند، که برای پشتیبان گیری از کامپیوترها و برای توزیع سورس کد و موارد دیگر استفاده می شود. Slackware هایی که tarball برای بسته های خود استفاده می کند حاوی اطلاعات خاص Slackware برای کمک به مدیریت بسته است. جنتو غیر عادی است زیرا سیستم بسته آن بر پایه کامپایل اکثر نرم افزارها از سورس کد است. این کار وقت گیر است اما مدیران با تجربه را قادر می سازد تا گزینه های کامپایل را برای بهینه سازی بسته ها برای محیط های سخت افزاری و نرم افزاری خود تغییر دهند.

نکته: تاربال ها مشابه فایل های فشرده رایج در ویندوز هستند. فصل 8، "Searching, Extracting and Archiving Data" نحوه ایجاد و استفاده از تاربال ها را شرح می دهد.

"Understanding Release Cycles": چرخه های انتشار را به زودی با جزئیات بیشتر در "Release Cycle" توضیح می دهیم. به عنوان یک قاعده کلی، توزیع هایی با چرخه های انتشار کوتاه، هدفشان ارائه جدیدترین نرم افزار ممکن است، در حالی که توزیع هایی با چرخه انتشار طولانی تر تلاش می کنند تا پایدارترین محیط های ممکن را فراهم کنند. برخی سعی می کنند آن را به هر دو صورت داشته باشند. برای مثال، اوبونتو نسخه های پشتیبانی بلندمدت (LTS) را در آوریل سال های زوج منتشر می کند. هدف دیگر انتشارات آن ارائه جدیدترین نرم افزار است.

جدول 1.1 تخمین شخصی ما از سطح مهارت مورد نیاز برای مدیریت توزیع را ارائه می دهد. همانطور که می بینید، ما اکثر توزیع های لینوکس را به عنوان نیاز به مهارت "متوسط" برای مدیریت توصیف کرده ایم. با این حال، برخی ابزارهای مدیریتی رابط کاربری گرافیکی کاربرپسند کمتری ارائه می کنند و بنابراین به مهارت بیشتری نیاز دارند. اوبونتو قصد دارد استفاده و مدیریت آن بسیار آسان باشد.

نکته: از طبقه بندی "متوسط" اکثر توزیع ها نترسید. هدف این کتاب کمک به شما در مدیریت ویژگی های اساسی چنین توزیع هایی است.

اکثر توزیع های لینوکس حداقل برای دو پلتفرم در دسترس هستند - یعنی انواع X (همچنین با نامهای A، 386IA32 و چندین گونه شناخته می شود) و 64x64 (همچنین به نامهای AMD64، EM64T، 64x64x 2007 کامپیوترهای 86 رایج ترین نوع بودند، اما اکنون کامپیوترهای 64x64 استاندارد شده اند. اگر یک کامپیوتر 64x64 دارید، می توانید یک توزیع 86x یا 64x64 را روی آن اجرا کنید، اگرچه دومی بهبود کمی در سرعت ایجاد می کند. پلتفرم های عجیب و غریب بیشتری مانند ARM (برای تبلت ها)، PowerPC، Alpha و SPARC در دسترس هستند. چنین پلتفرم هایی عمدتاً به سرورها و دستگاه های تخصصی محدود می شوند (به زودی توضیح داده خواهد شد).



### Real World Scenario

#### Which Distribution to Use?

با وجود انبوهی از توزیع های مختلف لینوکس، یکی از سوالات متداول برای کاربران تازه کار لینوکس این است که کدام یک را امتحان کنند. برخی از توزیع های لینوکس مانند Oracle و Red Hat و ارائه پشتیبانی پولی مشتری تمرکز دارند. این توزیع ها می توانند گران باشند، و اغلب نصب و استفاده از آنها برای کاربران مبتدی سخت است. نوع دوم توزیع های لینوکس، توزیع هایی هستند که برای کاربران پیشرفته لینوکس طراحی شده اند. توزیع های Slackware، CentOS و Gentoo Linux در این دسته قرار می گیرند. آنها انتظار دارند که بدانند چگونه اکثر نرم افزارها و سخت افزارها را خودتان نصب و پیکربندی کنید. کاربران پیشرفته لینوکس دوست دارند از این توزیع ها استفاده کنند زیرا می توانند دقیقاً آنچه را که باید روی سیستم نصب کنند سفارشی کنند.

نوع سوم توزیع‌های لینوکس، توزیع‌هایی هستند که برای کاربران تازه کار لینوکس طراحی شده‌اند. توزیع‌های محبوب فدورا، اوبونتو و مینت لینوکس در این دسته قرار می‌گیرند. نصب پیش‌فرض اکثر مسائل مربوط به پیکربندی نرم‌افزار و سخت‌افزاری را که باید نگران آن باشید برطرف می‌کند، و همگی ابزارهای گرافیکی کاربر پسند زیادی را برای انجام عملکردهای سرپرست، مانند افزودن حساب‌های کاربری، کاوش فضای دیسک، و کار با آنها ارائه می‌کنند. اتصالات شبکه. اگر در دنیای لینوکس مبتدی هستید، این نوع توزیع‌های لینوکس بهترین راه برای رفتن هستند.

## Understanding Release Cycles

جدول 1.1 چرخه‌های انتشار مورد استفاده تعدادی از توزیع‌های رایج لینوکس را خلاصه کرده است. مقدار های ذکر شده در جدول زمان بین انتشارها می‌باشد. برای مثال، نسخه‌های جدید اوبونتو دقیق هر شش ماه منتشر می‌شود. اکثر توزیع‌های دیگر برنامه زمانی ای منتشر می‌کنند که دارای "امکان مانور" می‌باشد؛ برای مثال، اگر زمان انتشار یک ماه به تعویق بیفتند قابل قبول است.

پس از انتشار یک توزیع، معمولاً تا مدتی بعد از انتشار نسخه بعدی پشتیبانی می‌شود—معمولاً چند ماه تا یک سال یا بیشتر. در طی این دوره پشتیبانی، نگهدارندگان توزیع برای باگ‌ها و مشکلات امنیتی آپدیت‌های نرم‌افزاری فراهم می‌کنند. بعد از اینکه دوره پشتیبانی به پایان برسد، شما می‌توانید به استفاده از توزیع ادامه بدھید، ولی دیگر خودتان تنها هستید—اگر به نرم افزار به روز نیاز دارید، باید خودتان از طریق سورس کد آن را کامپایل کنید یا امیدوار باشید که بتوانید یک پکیج باینری سازگار از منابع دیگر پیدا کنید. بصورت واقع بینانه، بطور کلی بهتر است که قبل از تمام شدن دوره پشتیبانی به آخرین نسخه بروزرسانی کنید. این موضوع باعث می‌شود تا توزیع‌هایی که چرخه انتشار طولانی تری دارند برای کسب و کارها خواهایند تر باشند از این رو که زمان طولانی بین نصب‌ها اختلال‌ها و هزینه‌های مربوط به بروزرسانی را به حداقل میرساند.

دو مورد از توزیع‌های جدول 1.1 (Gentoo و Arch) چرخه انتشارشان بصورت درجریان است. چنین توزیع‌هایی به صورت معمول عدد نسخه ندارند؛ در عوض، بروزرسانی‌ها به شکل درحال اجرا انجام می‌شوند، به همراه تمام دشواری‌هایی که ایجاد می‌کنند؛ هرچند شما گهگاهی باید یک بروزرسانی درهم گسیخته‌ای از یکی از زیرسیستم‌های بخصوصی انجام دهید، مثل بروزرسانی عمدۀ در محیط دسکتاپ.

قبل از انتشار یک نسخه جدید، اکثر توزیع‌کننده‌ها یک نسخه پیش از انتشار بوجود می‌آورند. نرم افزار آلفا بسیار نو و احتمالاً حاوی باگ‌های جدی است، در حالی که نرم افزار بتا پایدارتر است ولی با این وجود احتمال وجود باگ‌های بیشتری نسبت به نسخه نهایی دارد. به عنوان یک قانون کلی، شما بهتر است از استفاده از چنین نرم افزاری اجتناب کنید مگر اینکه بخواهید با گزارش باگ‌ها به پیشرفت توسعه کمک کنید یا به ناچار نیاز به امکانات جدید داشته باشید.

# Embedded Linux Systems

در کنار توزیع های فرآگیر PC، چندین توزیع لینوکسی دیگری وجود دارند که هدف تخصصی تری دارند. اصطلاح "سیستم تعییه شده" اجرای یک سیستم لینوکس کوچک که بر روی یک میکرو کامپیوتر مثل تلفن همراه یا دستگاه نظارتی را توصیف میکند. استفاده های عام از سیستم های تعییه شده لینوکسی عبارت اند از:

اندروید: بسیاری از گوشی های امروزی از یک سیستم عامل لینوکسی شناخته شده با نام اندروید استفاده میکنند که یوزر اینترفیس مشابهی نسبت به گوشی های دیگر دارد ولی در دل آن یک کرنل(هسته) لینوکسی و تعداد بسیار زیادی از زیرساخت های لینوکسی مشترک با PC قرار دارد.

نکته: اندروید بیشتر به عنوان یک سیستم عامل گوشی شناخته میشود ولی میتواند بر روی دستگاه های دیگر مثل تبلت ها یا کتابخوان های مجازی استفاده شود.

**تجهیزات شبکه:** بسیاری از روتر های پهنای باند، پرینت سرورها و باقی تجهیزاتی که به شبکه محلی برای اجرای کارهای مخصوص وصل میکنند از لینوکس استفاده میکنند. برخی اوقات شما میتوانید سیستم عامل استاندارد رو با یک سیستم عامل تغییر یافته برای اضافه کردن امکانات جدید جایگزین کنید. سیستم عامل های OpenWrt و Tomato دو مورد از این سیستم عامل های تغییر یافته لینوکسی هستند. البته چنین نرم افزارهایی رو بدون فکر نصب نکنید زیرا اگر به شکل غیر صحیح یا بر روی تجهیزات اشتباہ نصب شوند ممکن است باعث از کار افتادن و بلا استفاده شدن تجهیزات شوند.

**تجهیزات اینترنت اشیا:** در سال های اخیر اصطلاح اینترنت اشیا (IoT) در کلاس ها و اخبار موضوع داغی بوده است. اینترنت اشیا قصد دارد شبکه کوچکی از تجهیزاتی بسازد که قابلیت درک شرایط فیزیکی و کنترل سیستم ها را دارند. میکروپردازنده های کوچک با سیستم عامل های مخصوص به خود اطلاعات بدست آمده از سنسور ها مثل دما، رطوبت، نور یا حرکت را نظاره میکنند و از این اطلاعات برای کنترل موتورها، قفل ها و کلید ها استفاده میکنند. کنترل های تعییه شده ای مثل Arduino و Beaglebone و همچنین کنترل های بزرگتر و قوی تری مثل Raspberry Pi در حال تبدیل به موضوعات اشتیاق آور در مدارس و محیط های تولید هستند.

**دستگاه های TiVo:** این ویدیو رکوردر دیجیتال محبوب (DVR) از یک کرنل لینوکسی و تعداد زیادی از برنامه های پشتیبانی استاندارد به همراه درایور های اختصاصی و نرم افزار DVR استفاده میکند. اگرچه اکثر کسانی که از آن استفاده میکنند متوجه آن نیستند که آنها در باطن کامپیوتر های لینوکسی هستند.

اکثر سیستم های تعییه شده معمولاً نیازمند مقدار ناچیزی کار مدیریتی از طرف کاربران هستند حداقل نه در روشی که اینجور امور در کتاب توصیف شده. در عوض، این تجهیزات پیکربندی های ابتدایی را حل نموده و ابزارهای راه اندازی را برای کمک به کاربران بی تجربه برای وارد کردن تنظیمات حیاتی اولیه مثل تنظیمات شبکه و منطقه زمانی قرار دادند.

# Linux in the Cloud

فناوری ابری تا حد زیادی وجه دنیای کامپیوتری را دگرگون کرده است. انتقال منابع کامپیوتری و اپلیکیشن‌ها به یک محیط مشترک شبکه‌ای مقدار کمپانی‌هایی که با مشتریان تجارت و ارائه خدمات انجام میدهد را تغییر میدهد. لینوکس نقش مهمی در دنیای ابری ایفا می‌کند، بنابراین ایده خوبیست که مشخص کنیم که ابر چیست و چه نوعی از منابعی را فراهم می‌کند. این بخش پایه و اساس محاسبه ابری را در بر می‌گیرد.

## What is Cloud Computing?

اولین اشاره به اصطلاح ابر در مدارک مربوط به نسخه اصلی محیط شبکه‌ای ARPANET، پیشروی اینترنت امروزی، در سال 1977 بوده است. نماد ابر در آن مدارک عموماً به نمایندگی از شبکه بزرگی از سرورهای به هم متصل که از لحاظ جغرافیایی پراکنده بودند استفاده می‌شد. اگرچه در این محیط هر سرور مستقل و خودکفا بود و هیچ پردازش توزیع شده ای وجود نداشت.

اصطلاح رایانش ابری به پردازش توزیع شده مربوط است. در پردازش توزیع شده، منابع بین دو یا چند سرور به اشتراک گذاشته شدن تا یک امر واحد به انجام برسد مانند اجرا کردن یک اپلیکیشن. این محیط پیش رو در تبدیل به چیزی که ما امروزه به نام رایانش ابری می‌شناسیم شد و توسط کمپانی‌هایی مثل سرویس وب آمازون(AWS)، پلتفرم ابری گوگل و Microsoft Azure مشهور شد.

رایانش ابری قابلیت تحويلی منابع محاسباتی در سرتاسر اینترنت را فراهم می‌کند. امروز مشتریان می‌توانند منابع سخت افزاری و نرم افزاری مورد نیاز خود را از عرضه کنندگان رایانش ابری خریداری کنند. این شامل سرور‌ها، فضای ذخیره سازی، دیتابیس‌ها، شبکه‌ها، سیستم عامل‌ها و حتی اپلیکیشن‌های شخصی می‌باشد. شکل 1.4 سه شیوه فراهم کردن خدمات رایانش ابری را شرح میدهد.

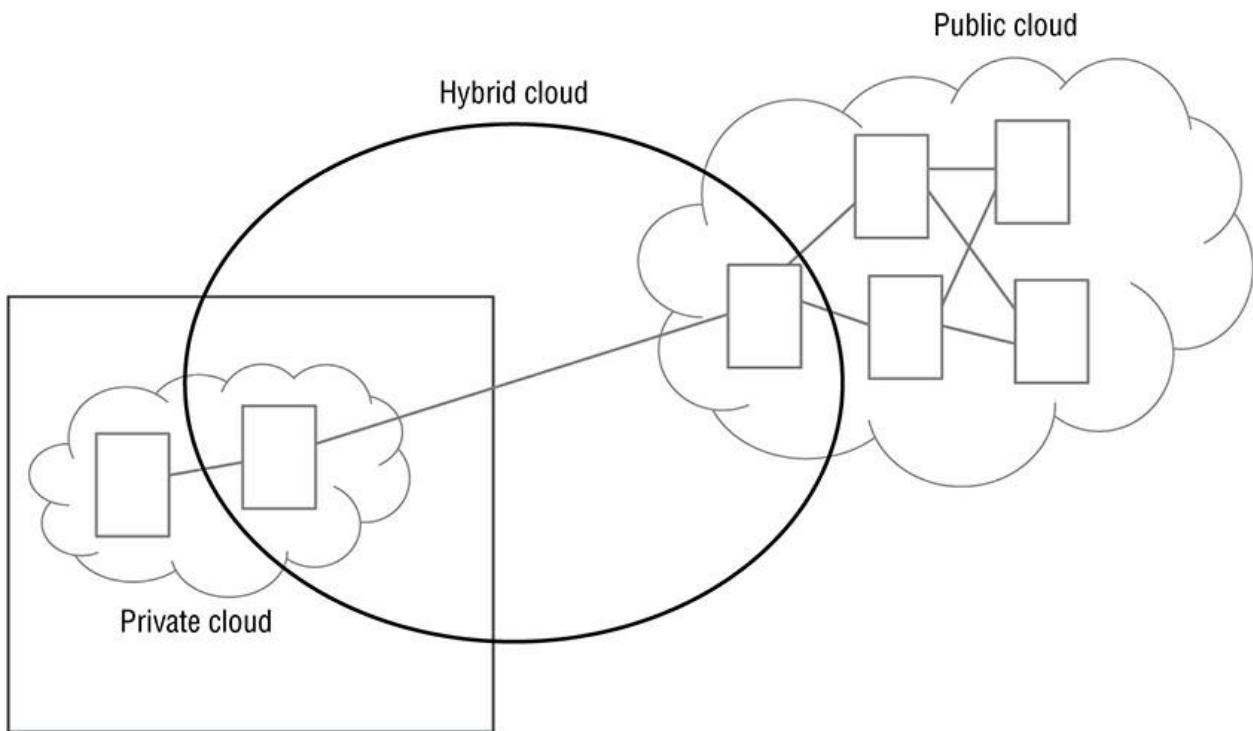


Figure 1.4 Cloud computing methods

همانطور که در شکل 1.4 میبینید، سه روش اصلی برای فراهم کردن محیط رایانش ابری وجود دارد:

**عمومی:** در محیط های عمومی رایانش ابری، یک شخص ثالث تمام منابع محاسباتی خارج از سازمان را فراهم میکند. این منابع معمولاً بین چندین سازمان به اشتراک گذاشته میشود.

**خصوصی:** در محیط خصوصی رایانش ابری، هر سازمان منابع رایانش ابری خاص خودش را میسازد تا به صورت داخلی منابعش را تامین کند.

**هیبریدی(ترکیبی):** در فضای ترکیبی رایانش ابری، منابع محاسباتی به صورت داخلی برای سازمان تامین میشوند ولی همچنان به یک فضای ابری عمومی خارجی برای تکمیل کردن منابع در زمان نیاز متصل میباشد.

## What are the Cloud Services?

محیط های رایانش ابری می تواند سطح منابع فراهم شده برای مشتریان را بسته به نیاز مشتری تغییر دهد. این بخش سه مورد از معروف ترین مدل های مورد استفاده برای تامین سطوح منابعی که شما از طریق فروشنده‌گان رایانش ابری پیدا می کنید را توضیح میدهد.

### Infrastructure as a Service (IaaS)

در مدل «زیرساخت به عنوان سرویس» عرضه کنندگان رایانش ابری منابع سروری سطح پایین برای میزبانی اپلیکیشن ها و سازمان ها فراهم میکنند. این منابع سطح پایین تمامی اجزای فیزیکی مورد نیاز شما برای یک

سرور فیزیکی مثل زمان، CPU، فضای ذخیره‌سازی و منابع شبکه را شامل می‌شود همانطور که در شکل 1.5 نشان داده شده.

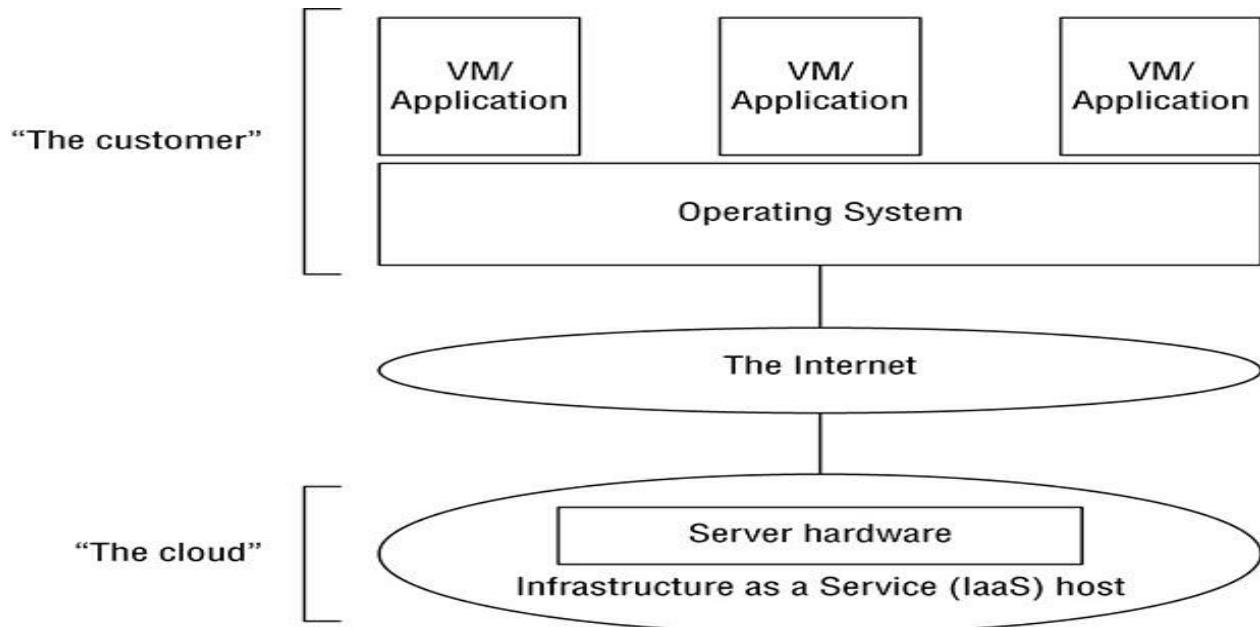


Figure 1.5 The IaaS cloud model

منابع سرور تامین شده ممکن است رو یک سرور واحد یا بین چندین سرور تقسیم شده باشد. در یک محیط تقسیم شده، سرور ها ممکن است در یک تسهیلات یا در چندین تسهیلات مختلف در شهر های متفاوت تفکیک شده باشند. این موضوع به افزایش دسترسی برای فراهم آوری کمک میکند.

همانطور که در شکل 1.5 نشان داده شد، در یک مدل IaaS، مشتری سیستم عامل و هر اپلیکیشنی که نیاز به اجرا دارد را تدارک میبیند. بیشتر محیط های IaaS بسیاری از سیستم عامل های مختلف از جمله لینوکس سرور و ویندوز سرور را پشتیبانی میکند. مشتری برای هرگونه کار مدیریتی که برای سیستم عامل و اپلیکیشن ها لازم است مسئول است و عرضه کننده رایانش ابری از زیرساخت محیط فیزیکی نگهداری میکند.

## Platform as a Service (PaaS)

در مدل «پلتفرم به عنوان سرویس» عرضه کننده همانطور که در شکل 1.6 نشان داده میشود هر دوی محیط فیزیکی سرور و محیط سیستم عامل را برای مشتری فراهم میکند.

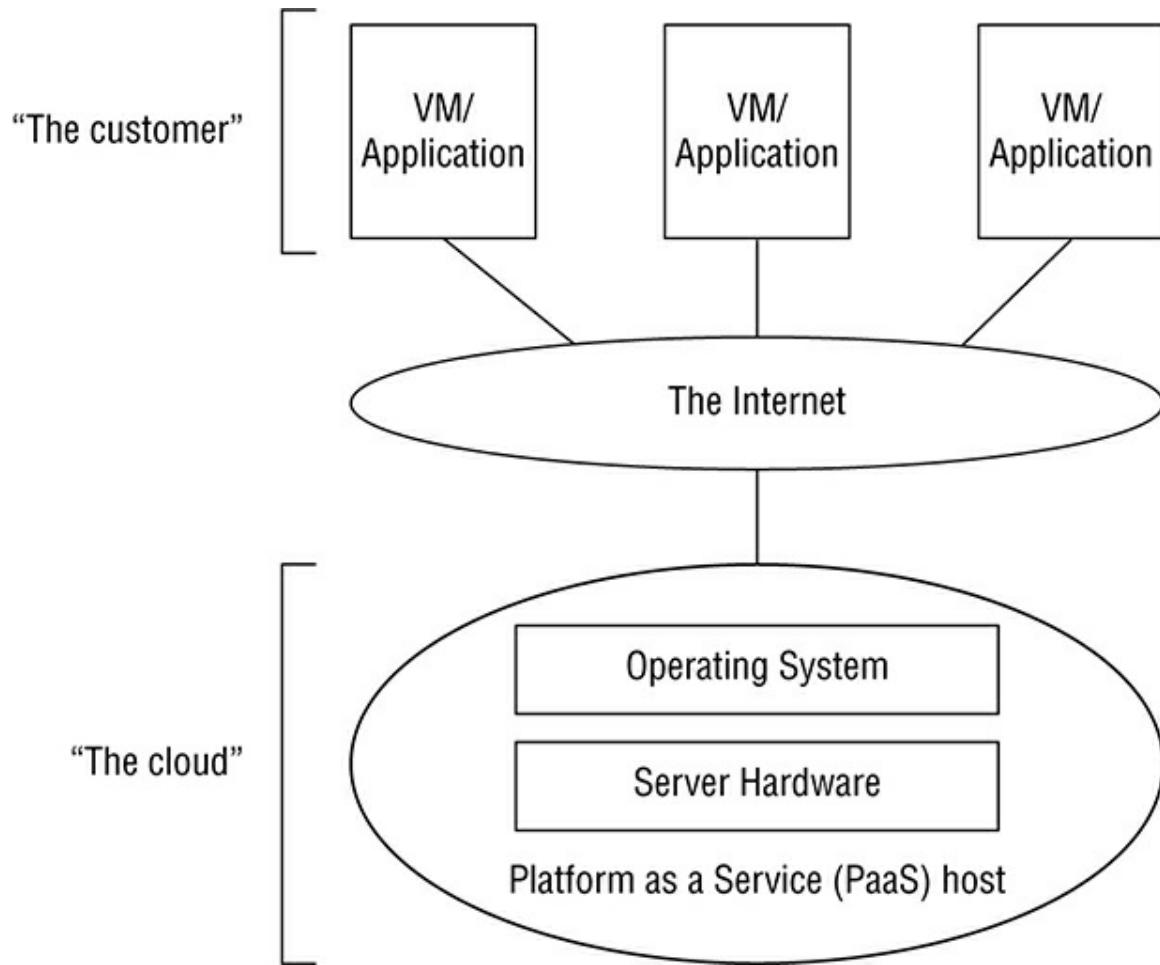


Figure 1.6 The PaaS cloud model

در مدل PaaS، عرضه کننده رایانش ابری مسئولیت اجزای فیزیکی و مدیریت سیستم عامل را به عهده میگیرد و با فراهم کردن پشتیبانی مدیریت سیستم از کامل بودن و به روز بودن سیستم عامل مطابق با آخرین انتشار و امکانات امنیتی اطمینان حاصل میکند. این باعث میشود که مشتری به طور کلی روی توسعه اپلیکیشن های اجرایی درون محیط PaaS تمرکز کند.

## Software as a Service (SaaS)

در مدل «نرم افزار به عنوان سرویس» عرضه کننده رایانش ابری یک محیط اپلیکیشنی کامل مثل سرور ایمیل، سرور دیتابیس یا سرور وب را تامین میکند. همانطور که در شکل 1.7 نشان داده شده، عرضه کننده محیط فیزیکی سرور، سیستم عامل و نرم افزار اپلیکیشن لازم برای اجرای عملکرد را فراهم میکند.

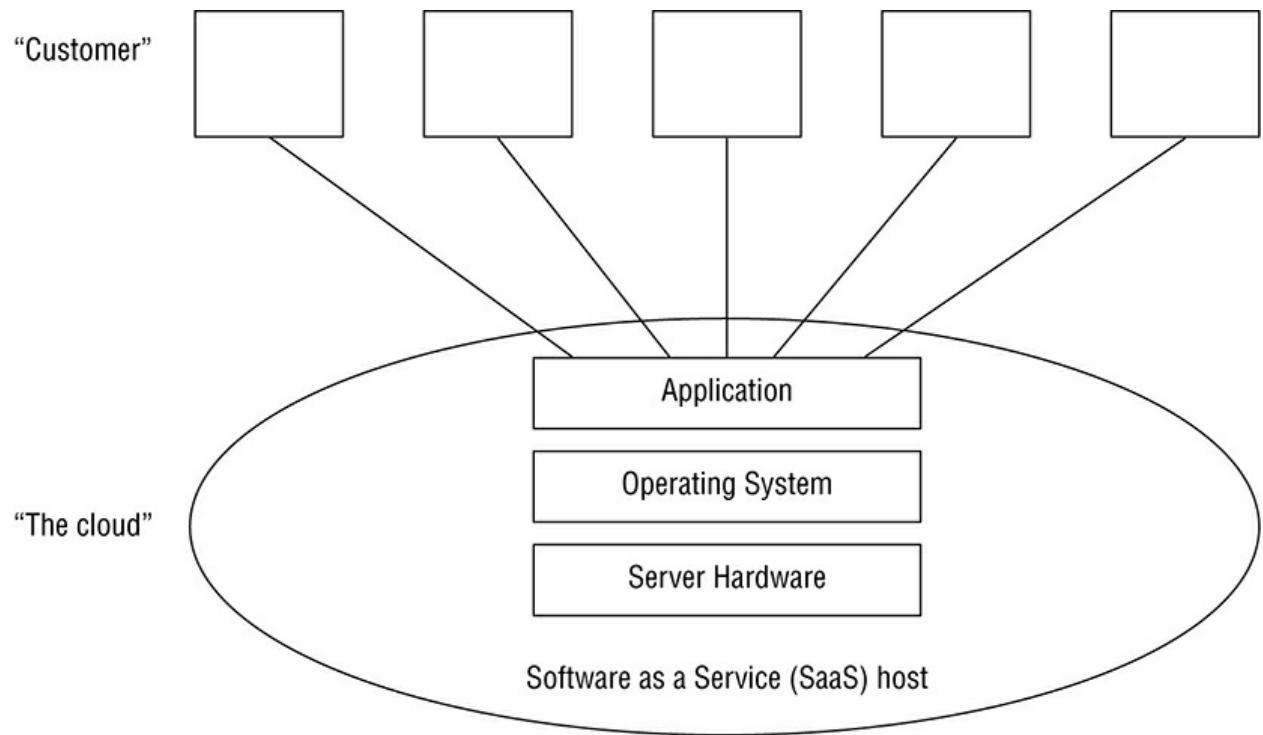


Figure 1.7 The SaaS cloud model

# CHAPTER 2

## Understandings Software Licensing

نرم افزار جزئی از انواع دارایی هایی است که توسط قانون کپی رایت و در برخی کشورها توسط قانون ثبت اختراعات کنترل میشود. به عنوان یک قانون کلی، این باعث میشود کپی کردن از نرم افزار در صورتی که مالکش نباشد غیرقانونی باشد. اما نرم افزار اپن سورس بر لایسنس ها تکیه می کند، که شرایط و ضوابطی که نرم افزار تحت آن منتشر میشود را دگرگون میسازد. همانطور که در این بخش توصیف می شود، لایسنس های اپن سورس حق و حقوق اضافه ای به کاربران نرم افزار اعطا می کند.

به طور کلی، نرم افزار اپن سورس به این سه سازمان مدیون است: سازمان نرم افزار آزاد (FSF)، سازمان پیشگامان اپن سورس (OSI) و CC (Creative Commons). هر سازمان یک فلسفه متفاوت و نقشی در دنیای اپن سورس دارد. همچنین تعداد بیشماری از لایسنس های مشخص اپن سورس وجود دارد که در آخر این بخش به همراه راه هایی که در تجارت استفاده می شوند خلاصه شدند.

### Investigating Software Licenses

قانون کپی رایت قرن هاست که وجود داشته است و مشخصا برای نرم افزار مخصوصا طرح نشده بوده است. با این وجود برای نرم افزار نیز صدق میکند. لایسنس هایی که خالقین برای نرم افزارهای خود اعمال میکنند با قانون کپی رایت در تعامل است تا حق و حقوق مشخصی که شما برای استفاده، تغییر و توزیع دارید و ندارید را بسازد. بنابراین شما باید اصول کلی را همانند تفاوت بین مالکیت اختصاصی و ضوابط لایسنس اپن سورس را بدانید.

### Exploring Copyright Protection and Software

کپی رایت همانطور که از نامش پیداست، یک حق قانونی رسمی برای ایجاد یک کپی از چیزی است. در اکثر کشورها، اگر یک کتاب بنویسید، عکسی بگیرید یا برنامه کامپیوتری بسازید، شما (فقط و فقط شما) حق ایجاد کپی از آن کتاب، عکس یا برنامه کامپیوتری دارید. اگرچه میتوانید به دیگران نیز حق ایجاد چنین کپی هایی را بدھید یا حتی کنترل کپی رایت را به کس دیگری اعطا کنید.

قوانين کپی رایت در هر کشوری متفاوت است اما اکثر کشورها عهدنامه برن را امضا کردند، یک قرارداد بین المللی که کشورها را مجبوب میکند قوانین کپی رایت یکدیگر را به رسمیت بشناسند. یعنی اگر کسی یک کتابی در ایالات متحده بنویسد(یا اپرا یا برنامه کامپیوتری)، آن اثر نه تنها در ایالات متحده، بلکه در ایسلند، کنیا، بریتانیا و هر کشور دیگری که قرارداد را تصویب کرند شامل کپی رایت میشود.

به دلیل اینکه اکثر قوانین کپی رایت قبل از به وجود آمدن کامپیوتر ها نوشته شدند، اغلب با نیاز های کامپیوتر ها جو نمی شوند. برای مثال، قانون کپی رایت کپی از آثار را قدغن میکند، اما یک برنامه کامپیوتری

بدون چنین کپی هایی عملاً بلا استفاده است. نمونه هایی از کپی هایی که لزوماً باید ایجاد شوند تا یک برنامه اجرا شود یا مقتضی برای امنیت باشد شامل موارد زیر است:

- یک کپی از برنامه نصب کننده در هارد درایو
- یک کپی از برنامه از دیسک درایو به RAM
- یک کپی از RAM در حافظه Swap
- یک کپی از RAM در فضای های کوچکتر و متنوعی در Cache یا بر روی مادربرد و یا CPU برای بهبود عملکرد
- یک یا چند بک آپ از دیسک درایو برای محافظت علیه خرابی دیسک

نکته: حافظه Swap یک نوع فضای دیسک است که به عنوان مکمل با RAM عمل میکند. برای مثال، اگر RAM پر شود، سیستم عامل از حافظه Swap مانند RAM استفاده میکند.

در گذشته، چنین کپی هایی بخاطر اصول استفاده منصفانه به طور کلی نادیده گرفته میشوند که به این معناست که استثناهایی برای کپی شدن بخشی از اثار کپی رایت ایجاد میشند. دیگر مثال هایی از استفاده منصفانه شامل نقل قول هایی در نقد و بررسی ها یا اخبار و گزیده هایی که در تحقیقات و آموزش استفاده میشوند میباشد. امروزه، قانون کپی رایت صریحاً نیاز کپی شدن نرم افزار برای استفاده از آن را به رسمیت میشناسد، البته حداقل در ایالات متحده که چنین است.



## Real World Scenario

### Patents, Trademarks, and Software

کپی رایت نمونه ای از دارایی معنوی است اما نمونه های بسیار دیگری هم وجود دارند. یکی از آنها حق انحصاری است. کپی رایت از یک اثر حفاظت میکند که میتواند یک جور اظهار ایده باشد، اما حق انحصاری از خود ایده نیز حفاظت میکند. حق انحصاری معمولاً شامل اختراعات میباشد.

در ایالات متحده، حق انحصاری نرم افزار مجاز است. اگرچه شما نمیتواند برای تمام یک برنامه حق انحصاری بگیرید، شما میتوانید برای الگوریتم هایی که برنامه استفاده میکند حق انحصاری بگیرید. چنین حقوق انحصاری ای هم معمول و هم بحث برانگیز هستند. برخی برنامه های اپن سورس از یک سری فرمت های فایل استفاده نمی کنند زیرا الگوریتم لازم برای استفاده از آنها حق انحصاری دارد و مالک کاربران غیرمجاز را تهدید به شکایت کرده است. متنقدین حقوق انحصاری نرم افزار ادعا میکنند چنین حقوقی مبتدل و بدیهی هستند، دو چیزی که یک اختراع فرضی نباید باشد. برخی اوقات کمپانی ها از حق انحصاری نرم افزار به عنوان راهی برای ممانعت از فروش یک محصول توسط دیگر کمپانی ها و یا طلب حق الزحمه استفاده میکنند.

در اکثر کشورهای دیگر، الگوریتم‌های نرم افزار نمیتواند شامل حق انحصاری شود. تلاش‌ها در حال انجام است تا رابطه بین نرم افزار و حقوق انحصاری عوض شود تا نرم افزارها در کشورهایی که این حقوق را ندارند شامل حقوق انحصاری شوند و در کشورهایی که امروزه حقوق انحصاری دارند حذف شوند.

نشان تجاری نوع دیگری از دارایی معنوی است. نشان تجاری یعنی نام‌ها، لوگو‌ها و دیگری شناسه‌های مشابهی که یک کالا یا کمپانی دارد. اغلب نرم افزارها و کمپانی‌هایی که آنها را تولید میکنند از نشان تجاری استفاده می‌کنند. در سال 1994 شخصی که ارتباط ناچیزی با جامعه لینوکس داشت نام لینوکس را به عنوان نشان تجاری خودش ثبت کرد و تقاضای حق الامتیاز کرد. بعد از دادخواهی، نشان تجاری به سازمان لینوکس (LMI) منتقل شد. ([WWW.LinuxFoundation.org/programs/legal/trademarks](http://WWW.LinuxFoundation.org/programs/legal/trademarks)).

به عنوان یک مصرف کننده نهایی، شما احتمالاً مجبور نیستید با حق انحصاری و نشان تجاری نرم افزار سروکله بزنید. مسائل حق انحصاری و نشان تجاری نرم افزار در سطح سازمانی اجرا می‌شوند. این با مسائل کپی رایت که میتواند اشخاص مختلف را تحت تاثیر قرار دهد در تضاد است. اگر شما برای کمپانی‌ای که نرم افزار منتشر میکنند کار میکنید قوانین حق انحصاری و نشان تجاری میتواند شما را تحت تاثیر قرار دهد. مخصوصاً اگر فکر میکنید که نرم افزار شما به طور بالقوه قوانین حق انحصاری یا نشان تجاری را زیر پا میگذارد، بهتر است با یک وکیل در این مورد مشورت کنید.

## Using Licenses to modify copyright terms

با اینکه نرم افزار تابع قانون کپی رایت است، بیشتر نرم افزار‌ها همراه یک لایسنس منتشر میشوند که اسناد قانونی ای هستند که حقوق، اعطای شده توسط کپی رایت را تغییر میدهند. در اکثر موارد، شما یک لایسنس را امضا نمی‌کنید، اگرچه در بعضی موارد شما یک دکمه را کلیک می‌کنید تا با ضوابط لایسنس موافقت کنید. در گذشته، لایسنس‌ها بر روی جعبه‌هایی که نرم افزارها در آنها توزیع میشدند چاپ میشدند. چنین لایسنس‌هایی معمولاً لایسنس قرارداد کاربر نهایی (EULAs)، لایسنس شرینک-رپ و لایسنس کلیک-رپ نام دارند. نرم افزار‌های اپن سурс نیز معمولاً با یک لایسنس در فایلی به نام COPYING عرضه میشوند.

نکته: دادگاه‌ها معمولاً از قابل اجرایی لایسنس‌های کلیک شدنی و مشابه حمایت کرده‌اند، اگرچه این نتایج جهانی نیستند.

لایسنس‌های نرم افزار میتوانند ضوابط کپی رایت را از طریق بیشتر یا کمتر کردن محدود کنندگی تغییر دهند. برای مثال، لایسنس جامع همگانی (GPL)، که لایسنس مورد استفاده کرنل لینوکس است به شما حقوق بازنشر نرم افزار، شامل سورس کد و کدهای باینری را میدهد. این نشان دهنده سبک شدن محدودیت‌های کپی رایت است.

به عنوان یک قانون کلی، لایسنس‌ها برای نرم افزار‌های انحصاری حقوق شما را تحت قانون کپی رایت محدود میکند، در حالی که لایسنس اپن سурс به شما حق و حقوق اضافی میدهد. اگرچه استثناهایی در این قانون میتواند وجود داشته باشد. برای مثال، لایسنس درون سازمانی قراردادی برای برنامه انحصاری‌ای است که به

یک سازمان حق ایجاد تعداد مشخصی از کپی را میدهد برای مثال 100 کپی از پردازشگر کلمات برای تمامی کامپیوتر های کمپانی.

## Looking at the free software foundation

بنیاد نرم افزار آزاد (FSF) عنصری حیاتی در دنیای اپن سورس است. پایه گذاری شده در سال 1985 توسط ریچارد استالمن، سازمان FSF عامل پشت پروژه گنو که در بخش قبلی توضیح داده شد می باشد. این سازمان دارای فلسفه خاصی می باشد که خود را در GPL، که لاینس برگزیده FSF است اظهار میکند و در ادامه توصیف می شود.

## Understanding the FSF Philosophy

بنیاد FSF از چیزی که نرم افزار آزاد مینامد حمایت میکند، که در مفهوم آزادی عمل در رابطه با نرم افزارها توصیف میشود. اصطلاح رایج «آزاد، نه مجانی» برای شفاف سازی این موضوع به کار میرود.

نکته: همانطور که FSF توصیف می کند، نرم افزار آزاد (Free Software) با نرم افزار رایگان متفاوت است. نرم افزار رایگان (Freeware) به طور کلی به معنای نرم افزاری است که بدون هیچ هزینه ای می توانید آن را تهیه کنید.

بنیاد FSF چهار آزادی بخصوص نرم افزار را به شرح زیر توصیف میکند:

- آزادی استفاده از نرم افزار برای هر مقصودی
- آزادی مشاهده و اصلاح منبع کد به دلخواه
- آزادی از تکثیر نرم افزار
- آزادی توزیع نسخه تغییر یافته خودتان

این آزادی ها به اصول OSI که به مختصر توصیف شد شباهت دارند. اگرچه تفاوت های مهمی در مفاد وجود دارد که در ادامه خواهید دید. FSF در [Gnu.Org/Philosophy/Free-sw.html](http://Gnu.Org/Philosophy/Free-sw.html) جزئیات بیشتری درمورد مفهوم این اصول ارائه میکند.

در یک دنیای ایده آل، طبق استانداردهای FSF، تمام نرم افزارها به صورت آزادانه به همراه منبع کد عرضه می شوند و تمام آزادی ها در پیرامون وجود دارد. برخی توزیع های لینوکس با همچین آرمانی مطابقت دارند، در حالی که باقی توزیع ها شامل نرم افزار های انحصاری میشوند. برخی اوقات، این نرم افزار رایگان است و باقی اوقات مقداری کد انحصاری است که به عرضه کنندگان امکان جلوگیری از تکثیر را میدهد و برای فروش نرم افزار هزینه ای طلب میکند. از آنجایی که نرم افزار آزادی لزوماً رایگان نیست، از دیدگاه FSF، فروش آن مشکلی ندارد. اگرچه، با توجه به آزادی های دیگر، قیمت نرم افزار آزاد به مرور زمان با دست به دست شدن به سمت صفر حرکت میکند.

هدف از تمام این صحبت ها نه تنها فقط توسعه دهنده ها و کمپانی ها، بلکه به کاربران نیز اختیار داده شود. اینکه شما میتوانید نرم افزاری که تقریباً کار مورد نیاز شما را انجام میدهد را به طوری تغییر

دهید که دقیقاً کار مورد نیاز را انجام دهد، نشان دهنده برتری نسبت به نرم افزار انحصاری است. اگر بتوانید نسخه تغییر یافته خودتان را عرضه کنید میتوانید به دیگران نیز کمک کنید (بر این فرض که عملکرد مشابهی نیاز داشته باشند). درنتیجه، اگر فلسفه FSF اجرا شود، میتواند به جامعه بزرگتری نفع برساند.

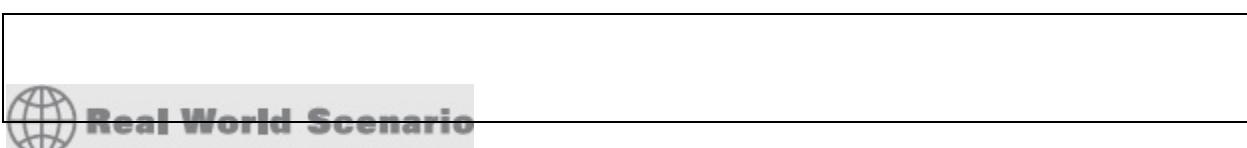
از فلسفه FSF و لایسنس هایی که از آن الهام می‌گیرند اغلب به عنوان *Copyleft* یاد میشود. این اصطلاح که از تغییر کلمه کپیرایت آمده، بازتاب این است که قوانین کپیرایت برای اطمینان از این است که آزادی دقیقاً در خلاف هدف اولیه خلق کپیرایت باشد و معنای تضمین آزادی کاربران بجای محدود کردن آنها برای ایجاد کپی از نرم افزار است. لایسنس های کپی‌لفت لازم میدانند که هرگونه نرم افزار تغییر یافته باید تحت ضوابط لایسنس نسخه اصلی منتشر شود. در نتیجه، از لایسنس کپی‌لفت اغلب به عنوان لایسنس دوچاره یاد میشود.

## Examining free software and the GPL

تعییر قانونی اصول FSF، در فرم GPL (اغلب به عنوان GNU GPL شناخته میشود) ناشی میشود. دو نسخه از GPL رایج هستند: نسخه 2 و نسخه 3. هر دو نسخه، چهار اصل آزادی فلسفه FSF را در نرم افزار لایسنس دار به کار میبرند و همچنین به صراحت اظهار میکنند که اعمال ثانویه ای باید تحت GPL صورت بگیرد، که در نتیجه آن را به یک لایسنس کپی‌لفت تبدیل میکند. این شرط از مصادره شدن یک برنامه اپن سورس توسط کمپانی جلوگیری میکند. برای مثال، بسیاری از کمپانی ها، توزیع های لینوکس تولید میکنند و برخی از کرنل های لینوکس که شامل پچ های اصلاح باگ هستند استفاده میکنند. این کرنل ها مانند کرنل رایج لینوکس، تحت GPL قابل دسترسی هستند. قانوناً هیچ کمپانی ای نمیتواند یک توزیع لینوکس بر پایه کرنل لینوکس پچ شده منتشر کند بدون آنکه پچ کرنل آن را فراهم کند.

نکته: یک توزیع لینوکس مجموعه ای از برنامه های متعددی است که هر کدام میتوانند از لایسنس های بخصوصی استفاده کنند. هیچ یک از لایسنس ها دارای ارجحیت بر دیگری نیستند.

نسخه 2 GPL در سال 1991 منتشر شد و سالها تسلط خود را حفظ کرد. در سال 2007، نسخه 3 با هدف اصلاح برخی ایرادات و مخصوصاً برای تغییر برخی قوانین نسخه 2 پیدار شد. نسخه 3 دارای شروط بخصوصی برای به چالش کشیدن محدودیت های سخت افزاری که چهار اصل آزادی FSF را محدود میکنند می باشد و به مشکلاتی مربوط به حق انحصاری نرم افزارها اشاره میکند. بسیاری از برنامه ها امروزه تحت ضوابط GPLv3 منتشر میشوند و بسیاری از برنامه های قدیمی امروزه از نسخه 3 به جای نسخه 2 استفاده میکنند. اگرچه برخی برنامه ها هنوز تغییری نکردند. قابل ذکر است که در میان اینها، خود کرنل لینوکس هنوز از نسخه 2 استفاده میکند. این تصمیم بسیار مهمی است زیرا به این معناست که کرنل لینوکس هنوزم میتواند به عنوان قلب دستگاه هایی که نسبتاً بسته هستند استفاده شود، مانند گوشی های اندروید. بسیاری از این دستگاه ها از پروسه بوت محدود کننده استفاده میکنند تا از بوت شدن کرنل های غیرمجاز جلوگیری کنند، پروسه ای که نسخه 3 آن را قدغن میسازد.



## Enforceable Legal Contract

شرکت Hancom به طور آزادانه نرم افزاری که تحت GNU GPL شرکت Artifex Software بود را بدست آورد و کد آن را تغییر داد. شرکت Hancom با قیمت گذاری بر روی آن نرم افزار، ضوابط لایسنس را نقض کرد. آرتیفیکس از هنکام شکایت کرد و هنکام در ادامه از دادگاه درخواست لغو پرونده را کرد و ادعا کرد که GNU GPL یک قرارداد لازم الاجرا نیست. قاضی دادگاه فدرال، ژاکلین اسکات کرلی، درخواست را رد کرد و قرارداد را لازم الاجرا دانست. دو کمپانی در آخر به توافق محرمانه ای رسیدند. (پرونده- No.16-cv-06982) (JSC)

نوع دیگر GPL، به عنوان GPL جزئی (LGPL) شناخته میشود. توسعه دهندهان معمولاً از LGPL برای کتابخانه ها که مجموعه ای از کدهای قابل استفاده برای برنامه های دیگر هستند، استفاده میکنند؛ برای مثال در لینوکس، کتابخانه ها امکاناتی را پیاده سازی میکنند که دیالوگ باکس ها و منو ها را ایجاد میکنند. بسیاری از برنامه های GUI از این امکانات استفاده میکنند و قرار دادن آنها در کتابخانه ها نه تنها به برنامه نویسان، بلکه به کاهش حجم برنامه ها نیز کمک میکند. با این حال، ضوابط GPL ملزم میکند که تمام برنامه هایی که از کتابخانه GPL دار استفاده میکنند تحت ضوابط GPL منتشر شوند. چنین التزامي سبب خلق GPL شده است، که به برنامه هایی که از کتابخانه های GPL استفاده میکنند این اجازه را میدهد که تحت لایسنس دیگری منتشر شوند، حتی لایسنس تجاری.

لایسنس مربوطه دیگری با نام لایسنس مستندات آزاد (FDL) وجود دارد که به جای نرم افزار، برای استفاده در اسناد و مدارک می باشد. به دلیل آنکه GPL برای نرم افزارها نوشته شده است، پس برای اسناد آماری به خوبی اعمال نمی شود پس گنو FDL را برای رفع اختلافات ایجاد کرد.

## Looking at the Open-Source Initiative

بروس پرنز و اریک ریموند سازمان OSI را در سال 1998 عموماً به عنوان یک سازمان جامع برای نرم افزار اپن سورس تاسیس نمودند. فلسفه آن که به زودی توصیف می شود به فلسفه FSF شباهت دارد ولی در جزئیات مهمی متفاوت است. به عنوان یک قانون کلی، نرم افزارهای بیشتری به عنوان اپن سورس اطلاق می شود تا آزاد (به طوری که FSF آن را تعریف میکند)، ولی این که چه چیزی به عنوان اپن سورس واجد شرایط می شود به معنای اپن سورس و به طور سخت گیرانه به این که OSI برحسب لایسنس آن چه چیز هایی را تایید کرده باشد.

## Understanding the Open-Source Philosophy

در دهه های 80 و 90 میلادی، جنبش نرم افزار آزاد به طرز چشمگیری در محفل های خاصی مانند دانشگاه ها و آکادمی ها و در میان علاقهمندان رشد کرد. با این حال، کمپانی ها در پذیرفتن نرم افزارهای آزاد کند بودند و

آنها بی که پذیرفتند، با اکراه یا ناآگاهانه چنین کردند. ادمین های سیستم که مجبور به انجام وظایف خود با بودجه کم بودند، به طور مخفیانه لینوکس، آپاچی، سامبا و دیگر نرم افزارهای آزاد را نصب میکردند تا از خرید نرم افزارهای جایگزین تجاری پیشگیری کنند.

تلاش های حمایتی FSF همواره بر اساس یک اصل اخلاقی بودند و هستند و نرم افزار از دید FSF باید آزاد باشد. چنین دیدگاهی برای برخی مردم خوشایند، و برای دیگران، مخصوصاً کمپانی ها که قصد درآمدزایی از فروش نرم افزار را دارند در بهترین حالت عجیب، و در بدترین حالت تهدید آمیز بنظر می آید.

برای چنین دلایلی، خالقین OSI سازمان خود را به شکلی طراحی کردند که حامی نرم افزار آزاد باشد با استفاده از اصطلاحی جدید به نام اپن سورس و با ملایم کردن اصول اخلاقی FSF، سازمان OSI هدف ترویج نرم افزار اپن سورس در دنیای تجاری را دارد. تفاوت در لحن نسبت به اصول FSF در شرح وظایف OSI در سایت OpenSource.org قابل مشاهده است. "اپن سورس دسترسی یک متود توسعه به نرم افزار می دهد. اپن سورس یک روش توسعه را برای نرم افزار فراهم میکند که از طریق آن بتوان قدرت بازبینی با دقت و شفافیت عمل را بدست آورد. اپن سورس قول کیفیت بالا، اطمینان بهتر، انعطاف پذیری بیشتر، هزینه کمتر و پایانی بر یغماگری عرضه کنندگان میدهد."

بزرگترین تفاوت فلسفی بین FSF و OSI در یکی از الزامات GPL منعکس میشود که آثار مشتق شده نیز باید تحت GPL منتشر شوند. سازمان OSI لاینسنス های بسیاری را از جمله GPL به عنوان اپن سورس تایید کرده است؛ با این حال، بسیاری از این لاینسنス ها کمبود محدودیت های مشابه دارند. نرم افزارهایی که در گذشته تحت این لاینسنス ها منتشر شده اند به نحوی تبدیل به کالای سورس بسته شده اند. سازمان OSI در چنین مسیری اعتراض نمی کند چرا که نرم افزار مذکور در موردی که مجاز بوده است تحت لاینسنス قرار گرفته. از طرفی دیگر FSF بهوضوح از چنین استفاده انحصاری در GPL خود ممانعت به عمل می آورد.

## :Permissive

لاینسنス های بسیاری توسط OSI تایید شده اند که لاینسنス اختیاری (Permissive) نام دارند. این لاینسنス ها که شامل لاینسنス آپاچی، لاینسنス های BSD و لاینسنス MIT (در این فصل به اینها پرداخته میشود) میشوند. همانند لاینسنス کپی لفت، یک لاینسنス اختیاری اجازه تکثیر، اشتراک گذاری و تغییر نرم افزار را تایید میکند. با این حال، لاینسن스 اختیاری به اندازه کپی لفت محدود کننده نمی باشد؛ از آنجایی که کپی لفت ملزم میداند که نرم افزارهای تغییر یافته مشتق شده از یک برنامه با لاینسنس، تحت ضوابط همان لاینسنス منتشر شود. لاینسنس اختیاری حداقل ملزومات را در رابطه با تکثیر نرم افزار دارد. برخی لاینسنス ها که به عنوان لاینسنس اختیاری محسوب می شوند اجازه اضافه کردن ضوابط اضافی یا متفاوت از لاینسنス اصلی را برای نرم افزار مجاز میدانند. بنابراین، لاینسنس اختیاری گاهی به عنوان لاینسنス یک جانبی نیز شناخته میشود.

امروزه برخی تنیش ها میان طرفداران نرم افزار آزاد در تعریف FSF و فعالان جامعه اپن سورس وجود دارد. اگرچه اکثر موقع هر دو یک هدف به اندازه کافی مشابه را دنبال میکنند به طوری اختلافات میان آن ناچیز محسوب میشود. در واقع، دو اصطلاح نرم افزار رایگان و اپن سورس (FOSS) و نرم افزار آزاد و اپن سورس (FLOSS) برخی اوقات به عنوان یک اصطلاح کلی برای اشاره صریح به دو نوع نرم افزار و توسعه مورد استفاده قرار میگیرند.

## Defining Open-Source Software

تعریف اپن سورس در [Opensource.org/Definition](http://opensource.org/Definition) قابل مشاهده است و شامل 10 اصل می شود که در زیر توصیف شدند:

**تکثیر آزاد:** اجازه تکثیر و مخصوصا تکثیر انبوه باید در لایسنس تایید شده باشد.

**دسترسی سورس کد:** خالق باید سورس کد و کدهای باینری(اگر لازم و شدنی باشد) را فراهم و اجازه تکثیر آن را صادر کند.

**اجازه مشتق شدن:** لایسنس باید اجازه تغییر نرم افزار و عرضه چنین تغییراتی را تحت لایسنس اصلی به دیگران اعطا کند.

**نکته:** معنای اپن سورس اجازه این را میدهد که لایسنس، تکثیر را تحت لایسنس اورجینال لازم بداند و اجراء نمیکند.

**احترام به صداقت اپن سورس:** ممکن است لایسنس از تکثیر سورس کد تغییر یافته جلوگیری کند اما فقط زمانی که فایل های پچ در کنار سورس کد اصلی منتشر شوند. این لایسنس احتمالا ملزم میکند که نام یا ورژن نسخه تغییر یافته عوض شود.

**عدم تبعیض علیه اشخاص یا گروه خاص:** لایسنس نباید نسبت به هیچ شخص و گروهی از مردم تبعیض قائل شود.

**عدم تبعیض علیه حوزه های مختلف:** لایسنس نباید استفاده از نرم افزار را در هر حوزه ای مثل تجارت یا تحقیقات ژنتیکی منع کند.

**تکثیر خودکار لایسنس:** لایسنس باید شامل هرکسی که نرم افزار را دریافت میکند باشد بدون نیاز به قرارداد جدآگانه.

**عدم اختصاصی بودن محصول:** لایسنس نباید ملزم کند که نرم افزار به عنوان بخشی از یک برنامه بزرگتر عرضه و مورد استفاده قرار گیرد. یعنی شما میتوانید یک برنامه از یک مجموعه بزرگتر را استخراج و جدآگانه تکثیر کنید.

**عدم محدودیت بر روی دیگر نرم افزارها:** لایسنس نباید بر روی نرم افزاری که همراه با نرم افزار لایسنس شده عرضه می شود محدودیت اعمال کند.

**بی طرفی تکنولوژی:** لایسنس نباید بر اساس اینترفیس یا تکنولوژی خاصی محدود شود.

**نکته:** 10 اصل OSI از اصول اظهار شده توسط توسعه دهندهان دیبان گنو/ توسعه دهندهان لینوکس مشتق می شود.

سه اصل اول این اصول حداقل در بحث درک هدف تکنولوژی اپن سورس، از باقی اصول مهم تر می باشند. مجموعه به طور کلی شباهت زیادی به چهار اصل FSF و توضیحات اضافی راجع به پیامدهای آن در صفحه وب FSF دارد. ([Gnu.org/philosophy/free-sw.html](http://Gnu.org/philosophy/free-sw.html)) همانطور که توضیح داده شد، تفاوت هایی وجود دارد مخصوصاً رعایت استلزمات نسخه های مشتق شده.

## Looking at the Creative Commons

از آنجاییکه FSF و OSI هر دو متعهد به ترویج آزادی های نرم افزاری هستند، اهداف کمپانی Creative Commons (وب سایت [CreativeCommons.org](http://CreativeCommons.org)) گستردۀ تر است. لاینس های آن مخصوص صدای های ضبط شده، ویدیوها، آثار متنی و غیره است و لزوماً مخصوص برنامه های کامپیوتری نیست. اگرچه این سازمان به ترویج نوعی از آزادی که مدنظر OSI و FSF می باشد کمک می کند.

سازمان Creative Commons توسط لارنس لسیگ تاسیس شد. هدف آن مبارزه با چیزی است که خالقین و حامیانش آن را به عنوان فرهنگ خلاقیت میشناسند که به طور فزاینده به مجوزهای تایید شده (یا تایید نشده) توسط کسانی که حق کپیرایت آثار اولیه را در اختیار دارند مربوط است.

اکثر فرهنگ امروزی از آثار ابتدایی نشات گرفته است. برای مثال، مجموعه فیلم های جنگ ستارگان از افسانه ها و اساطیر رایج الهام گرفته است. با این حال، جنگ ستارگان به خودی خود کپیرایت شده است که باعث میشود حق و حقوق هنرمندان کنونی را برای آثاری که از آن بدون مجوز مشتق شده است را محدود سازد. کمپانی Creative Commons اهداف خود را با فراهم کردن لاینس هایی ترویج میدهد که به خالقین کمک میکنند آثار خود را حفظ کنند ولی در عین حال به دیگران اجازه کپی، تکثیر و استفاده غیر تجاری از آثار اصلی را به طور آزادانه را بدهد.

مجموعه لاینس های Creative Commons از شش لاینس که برای اهداف مختلف طراحی شدند تشکیل شده است. شما میتوانید با پاسخ دادن به تعدادی سوال در وبسایت [CreativeCommons.org/choose](http://CreativeCommons.org/choose)، یک لاینس را انتخاب کنید برای مثال اینکه آیا میخواهید اجازه استفاده تجاری از اثر خود را تایید کنید یا خیر.

## Using Open-Source Software

به عنوان یک کاربر فردی، شما احتمالاً نیاز به کاوش عمیق در جزئیات اپن سورس را ندارید. اصول پشت مقررات OSI تضمین میکند که شما حق استفاده و حتی تکثیر برنامه های اپن سورس را هرگونه که صلاح بدانید دارید؛ البته اگر شما در حال ساخت یک کسب و کار هستید، مخصوصاً تجاری که مربوط به ساخت یا تکثیر نرم افزار اپن سورس باشد، در آن صورت شما نیاز به شناخت بهتر این لاینس ها دارید. بنابراین، این بخش چند مورد از آنها را با جزئیات بیشتر به همراه روش هایی که کمپانی ها از لاینس های اپن سورس در مدل های خود استفاده میکنند را توضیح میدهد.

## Understanding Open-Source License

تمام لایسنس های اپن سورس خصوصیات خاص خودشان را دارند. اکثر اینها مورد توجه توسعه دهنده‌گانی قرار دارند که قصد شرکت در یک پروژه نرم افزاری دارند، ولی در برخی مواقع میتوانند برای یک ادمین سیستم نیز مهم باشند. لایسنس های اصلی اپن سورس شامل موارد زیر می باشند:

**LGPL و GNU GPL :** همانطور که قبل تر اشاره شد، کرنل لینوکس از GPL نسخه 2 استفاده می کند، و بسیاری از ابزار های دیگر لینوکس از GPL استفاده میکنند (یا نسخه دوم یا نسخه سوم). بسیاری از کتابخانه های لینوکس از LGPL استفاده میکنند.

**BSD :** لایسنس های توزیع نرم افزاری برکلی (BSD) توسط سیستم عامل های اپن سورس BSD و نرم افزار های مختلفی که برای آنها ساخته شده اند استفاده میشود. برخلاف GPL، لایسنس های BSD اجازه میدهد که نسخه های تغییر یافته تحت لایسنس های دیگری منتشر شوند. آخرین نسخه های این لایسنس به لایسنس MIT مقداری شباهت دارند.

نکته: دو لایسنس BSD رایج هستند: لایسنس های نسخه Three-Clause و Two-Clause. برخی اوقات لایسنس Two-Clause ساده شده یا FreeBSD نامیده می شود. نسخه Three-Clause در اشاره به نسخه قدیمی تر (Four-Clause) گاهی نسخه جدید یا اصلاح شده نامیده میشود.

**MIT :** موسسه تکنولوژی ماساچوست (MIT) نیروی اصلی پشت Window System X بود (X به اختصار)، و لایسنس MIT همچنان برای Xorg 11 استفاده میشود؛ کاربردی از X که همچنان با چندین توزیع مختلف لینوکس همراه است. با این حال، برخی توزیع های لینوکس از X به Wayland که آن هم از لایسنس MIT استفاده میکند کوچ کرده اند. لایسنس MIT به طور فوق العاده ای کوتاه است و حدود 160 کلمه می باشد.

**Apache :** همچون لایسنس های BSD و MIT، لایسنس آپاچی یک لایسنس اپن سورس است که امکان تکثیر تحت لایسنس خود و دیگر لایسنس ها را فراهم میکند. اگر یک فایل متنی با نام NOTICE همراه با اثر اصلی عرضه شود باید در هرگونه آثار مشتق شده از آن قرار گیرد. این به توسعه دهنده اصلی این امکان را میدهد که اطلاعات تماس و دیگر اطلاعات را فراهم کند، حتی برای کاربرانی که از نسخه های تغییر یافته استفاده میکنند.

نسخه اصلی لایسنس آپاچی (نسخه 1.0) در سال 1995 ایجاد شد. نسخه فعلی 2.0 می باشد و بنابراین گاهی اوقات لایسنس آپاچی با نام آپاچی 2 شناخته می شود.

نکته: همانطور که از نامش پیداست، لایسنس آپاچی از مرورگر وب آپاچی نشات می گیرد؛ با این حال، برای بسیاری از پروژه های دیگر نیز استفاده میشود.

لایسنس های بسیار دیگری نیز توقعات OSI را برآورده میکنند. شما میتوانید یک لیست کامل از آنها را در وبسایت OpenSource.org/Licenses مشاهده کنید.

جزئیات لایسنس های اپن سورس مختلف احتمالا برای اکثر سیستم ادمین ها فاقد اهمیت است. شما میتوانید هر طور که صلاح می دانید از نرم افزار اپن سورس استفاده کنید و آن را تکثیر کنید. اگرچه اگر یک برنامه را تغییر دهید، باید از ملزمات تکثیر آن آگاه باشید، مخصوصا اگر قصد ادغام دو یا چند برنامه یا

انتشار یک برنامه تحت یک لایسنس تغییر یافته را داشته باشید. شما همچنین باید از این موضوع آگاه باشید که برخی توزیع های لینوکس ممکن است شامل نرم افزارهایی باشند که اپن سورس به حساب نیایند. برخی از آنها نرم افزار های تجاری هستند و برخی دیگر در دسته های دیگر قرار می گیرند.

**هشدار:** برخی ترکیب های لایسنس های اپن سورس با یکدیگر ناسازگار هستند، و این به این معناست که شما قانوناً نمی توانید کدها را باهم ادغام کنید و نسخه تغییر یافته را منتشر کنید.

یک مورد آخر در رابطه با توصیف لایسنس های نرم افزار، لایسنس لینوکس به طور کلی است. زمانی که شما یک فایل image دانلود میکنید یا یک پکیج لینوکس خریداری میکنید، نرم افزاری که شما کسب میکنید از لایسنس های متنوعی استفاده میکند؛ مثل GPL، BSD، MIT و غیره. اکثر این لایسنس ها اپن سورس هستند، اما نه همه آنها. همراه با بسیاری از توزیع ها، تعدادی اشتراک افزار (Shareware) و پکیج های نه چندان اپن سورس عرضه میشود؛ مانند برنامه اشتراک افزار گرافیکی XV. پکیج های جزئی گاهی اوقات به طور آشکارا دارای نرم افزار های تجاری می باشد. به همین دلیل، اگر شما یک دیسک پکیج لینوکس خریداری کردید، در صورتی که راجع به موضوع تحقیق نکرده اید و از بلامانع بودن تکثیر آن مطمئن نشده اید نباید آن را کپی کنید. اگر عرضه کنندگان لینک دانلود رایگان فراهم کنند، کپی از آن بلامانع است.

توزیع های لینوکس شامل برنامه های نصب کننده، برنامه های پیکربندی و غیره می باشد. این ابزارها معمولاً تمام چیزی هستند که مالکین پکیج ها میتوانند بر آن ادعای کپیرایت داشته باشند. اکثر نگهدارندگان توزیع ها روتین نصب و پیکربندی های خود را تحت لایسنس GPL و برخی لایسنس های اپن سورس دیگر فراهم کرده اند، اما این مورد همیشگی و ثابت نیست. چنین جزئیاتی میتواند چیزی که یک سیستم عامل اپن سورس بنظر می آید را به چیز دیگری که عمل اپن سورس نیست تبدیل کند. دلیل این از سیاست استفاده مطلق از نرم افزار های اپن سورس در پکیج های اصلی خود پشتیبانی میکند، با این حال از نرم افزار های قابل تکثیر غیر اپن سورس در پکیج های "غیر رایگان" خود استفاده میکند.

از آنجایی که یک توزیع کامل لینوکس از اجزایی تشکیل شده است که لایسنس های زیادی استفاده میکنند، درست نیست که از اعمال یک لایسنس یا یک کپیرایت برای تمام سیستم عامل صحبت کنیم. در عوض، بهتر است توزیع های لینوکس را به عنوان یک مجموعه ای از محصولات که با ابزار یکپارچه کردن نصب عرضه میشود در نظر بگیریم. اگرچه تعداد گسترده ای از برنامه ها تحت یک لایسنس اپن سورس واحد عرضه میشوند.

## Understanding Open-Source Business Models

برخی توزیع های لینوکس مثل دل، توسط افراد یا سازمان های داوطلب نگهداری میشود. باقی توزیع ها مثل لینوکس تجاری Red Hat، توسط کمپانی های سودجو نگهداری میشوند. حال چگونه یک کمپانی میتواند از چیزی که هسته محصول آن رایگان در اینترنت وجود دارد سود کسب کند؟ چندین روش برای کسب درآمد از نرم افزار اپن سورس وجود دارد که عبارت اند از:

**سرویس ها و پشتیبانی:** محصول به خودی خود میتواند اپن سورس باشد و حتی رایگان در اختیار عموم قرار بگیرد و در عین حال کمپانی سرویس خدمات دهی مثل آموزش و خطوط تلفنی پشتیبانی تکنیکال را با هزینه

تامین کند. برای مثال، یک بازی ویدیویی می‌تواند اپن سورس باشد ولی نیاز به عضویت در یک سرویس آنلاین برای ارائه امکانات کامل داشته باشد.

**لایسنس دوگانه:** یک کمپانی میتواند دو نسخه از یک محصول را تولید کند: یکی به صورت اپن سورس قابل دسترس باشد و دیگری دارای امکاناتی باشد که در نسخه اپن سورس وجود ندارد. نسخه اپن سورس به نمونه های تستی در سوپرمارکت ها شباهت دارد و روشی برای جذب مشتری برای نسخه تجاری است.

**محصولات چندگانه:** محصول اپن سورس میتواند فقط یکی از محصولات کمپانی باشد و درآمد از خط تولید های دیگری مثل نرم افزار های دیگر یا محصولات مشابه مانند دستورالعمل ها حاصل شود.

**درایور های اپن سورس:** یک مورد خاص برای فروشندگان سخت افزار. آنها میتوانند درایورها یا اپلیکیشن های اپن سورس مخصوص سخت افزارهای خاص به عنوان راهی برای ترویج سخت افزارهای خود تولید کنند.

**نکته:** زمانی که عرضه کنندگان یک درایور اپن سورس منتشر میکنند، در کدنویسی آن اطلاعاتی راجع به سخت افزار نمایان می شود بنابراین برخی عرضه کنندگان نسبت به انتشار درایور اپن سورس بی میل هستند.

**پاداش ها:** پاداش ها یک روش پرداخت جمعی می باشد. کاربران با خرید یک نرم افزار یا امکانات جدید می توانند از آثار اپن سورس استفاده کنند. وبسایت هایی مثل (BountySource.com) و (FossFactory.org) میتوانند در گردhem آوری کاربران کمک کنند که هر کدام از آنها ممکن است به تنها ی قادر به اهدای پول کافی برای انگیزه دادن به توسعه دهندهان برای برنامه نویسی را نداشته باشد. با پاداش ها، برنامه نویسی که پروژه را سریعتر از دیگران تمام کند اجازه دریافت پول های جمع شده برای پروژه به او داده میشود.

**اهدایی ها:** بسیاری از پروژه های اپن سورس از کمک هزینه های اهدایی برای توسعه های خود استقبال میکنند. اگرچه این یک مدل تجاری نیست، اما در تأمین هزینه عملیات های سازمان هایی مانند FSF کمک میکند.

فرای این فرصت های تجاری، بسیاری از نرم افزارهای اپن سورس در آکادمی ها توسط دولت ها، سازمان های داطلب، علاقمندان و... توسعه داده میشوند. حتی کمپانی ها میتوانند تحریک به بازگرداندن تغییراتی که برای خود انجام داده اند شوند؛ زیرا احتکار تغییرات باعث افزایش امور داخلی برای کمپانی میشود- اگر یک تغییر داخلی به خالق اصلی بازگردانده نشود، کمپانی مجبور به اعمال دوباره تغییرات برای هر انتشار های جدید نرم افزار میشود.

# Chapter 3

## Investigating Linux's Principles and Philosophy

شما میتوانید مرتبًاً یک محصول یا تکنولوژی را صرفاً بر اساس عمل گرایی انتخاب کنید - کدام سیستم عامل برای انجام یک عملیات به خوبی عمل میکند، چه مجموعه نرم افزاری کمترین هزینه را دارد و غیره. اگرچه برخی اوقات، دانستن فلسفه و اصولی که پشت یک تکنولوژی قرار دارد میتواند کارآمد باشد و حتی شما را در انتخاباتان راهنمایی کند.

این برخی کاربران لینوکس صحیح است؛ مدل اپن سورس لینوکس که در فصل اول، "Selecting an Operating System" معرفی کردیم پیامدهایی دارد که میتواند نحوه عملکرد لینوکس را تحت تاثیر قرار دهد. علاوه بر این، برخی در دنیای لینوکس در رابطه با این اصول بسیار احساساتی و پرهیجان میشوند. چه با این افراد موافق باشید چه نباشید، درک دیدگاه آنها که در محل کار، شبکه های اجتماعی، کنفرانس ها و ... پیدا میشود میتواند به شما در درک و تحسین فرهنگ لینوکس کمک کند.

این فصل این موضوعات را با شروع از موضوع خاستگاه لینوکس و توسعه آن طی زمانه بررسی میکند. پس از آن اصول اپن سورس و چگونگی تاثیر آنها بر نحوه عملکرد یک سیستم عامل اپن سورس در دنیای واقعی را توضیح میدهیم. در آخر، برخی از نقش هایی که لینوکس در آنها به عنوان سیستم عامل تعییه شده، سیستم عامل دسکتاپ یا لپ تاپ و یا سیستم عامل سرور کار میکند را توضیح میدهیم.

## Linux Through the Ages

با وجود اینکه بنا بر استاندارد های تاریخی، زمان تولد لینوکس در سال 1991، اخیرا محسوب میشود، در دنیای کامپیوتری 32 سال یک عمر محسوب میشود. با این وجود، نرم افزار و فرهنگ آن در اوایل دهه 90 میلادی و حتی قبل از آن، میراث بزرگی و مهمی را با خود به دنیای امروزی حمل کرده است. چون بهر حال چیزهایی که ما امروزه استفاده میکنیم بر روی پایه های چیزی که در گذشته ساخته شده بنا شده است. بنابراین، با نگاه به چگونگی سرچشمه گرفتن لینوکس به شما در درک لینوکس امروزی کمک میکند.

## Understanding Linux's Origins

در سال 1991، همانند امروز، کامپیوتراها بر اساس اندازه و توانایی ها طبقه بندی میشوند. کامپیوتراها ممکن به هر کدام از دسته ها تعلق داشته باشند، از کامپیوتر شخصی دسکتاپ (PC) گرفته تا ابر کامپیوتراها. کامپیوتراهای x86 که از اجداد مستقیم کامپیوتراهای امروزی محسوب میشوند بازار کامپیوتری 1991 را فتح کردند؛ اگرچه انواع دیگری از کامپیوتراها مثل مک ها قابل دسترس بودند. چنین کامپیوتراهایی به طور کلی از CPU های متفاوتی استفاده می کردند و سیستم عامل های خاص خود را اجرا میکردند.

نکته: کامپیوتراهای امروزی میتوانند به همان روش 1991 طبقه بندی شوند، با این حال برخی جزئیات تغییر کرده است. یک تغییر قابل توجه کامپیوتراهای تعییه شده (Embedded Computers) یا گوشی هوشمند است.

در سال 1991، اکثر کامپیوتراها از سیستم عامل مایکروسافت دیسک (DOS MS-DOS, PC-DOS) یا استفاده میکردند. داس از لحاظ استاندارد های امروزی بسیار محدود بود؛ یک سیستم عامل تک کاره ( فقط قابلیت اجرای یک برنامه همزمان را داشت) بود که حتی از تمام ظرفیت CPU و RAM استفاده نمیکرد. نسخه های قابل دسترس مایکروسافت ویندوز در 1991 بر روی داس اجرا میشد. با اینکه نسخه های ابتدایی ویندوز به برخی از محدودیت های داس کمک کرد اما به طور بالفعل آنها را اصلاح نکرد. این نسخه های ابتدایی ویندوز از چندکاره بودن مشارکتی (Cooperative Multitasking) استفاده کرد؛ برای مثال، برنامه میتوانستند ظرفیت CPU را داوطلبانه به دیگر پروسه ها اهدا کنند. کرنل داس قادر به سلب اختیار از برنامه ای که ظرفیت CPU را اختیار گرفته نبود.

بالاتر از سطح کامپیوترا دسکتاپ، یونیکس یک سیستم عامل رایج در 1991 بود. در مقایسه با داس و نسخه های دیگر ویندوز در آن زمان، یونیکس یک سیستم عامل امروزی تر بود. یونیکس از چندین اکانت پشتیبانی میکرد و چندکاره بودن حقیقی را فراهم میکرد به طوری که کرنل آن میتوانست ظرفیت CPU را برای برنامه های زمانبندی کند حتی اگر برنامه ها حاضر به رها کردن کنترل نبودند. این امکانات، ضروریات کاربردی ای برای بسیاری از سرورها و کامپیوتراهای چند کاربره مثل مینی کامپیوتراها و کامپیوتراهای مرکزی بودند.

نکته: یونیکس تنها سیستم عامل چند کاربره و چندکاره در سال 1991 نبود. باقی سیستم عامل‌ها مانند (Virtual Memory System) VMS نیز وجود داشتند. با این حال یونیکس بیشترین نسبت را به تاریخ لینوکس دارد.

با گذشت زمان، توانایی‌های هر دسته از کامپیوترها رشد کرد. با اکثر سنجش‌ها، کامپیوترهای دسکتاپ امروزی قدرت مینی کامپیوترها و حتی کامپیوترهای مرکزی 1991 را دارا هستند. سیستم عامل‌هایی که در 1991 در کامپیوترها استفاده می‌شد به خوبی با سخت افزارهای قوی جور نمی‌شدند؛ امروزه کامپیوترهای دسکتاپ قدرت کافی برای اجرای بهترین سیستم عامل‌های 1991 را دارند. به همین دلیل، داس و کامپیوترهای کوچکش به طور عمدۀ به نفع یونیکس و جایگزین‌های دیگر کنار کشیدند.

نسخه‌های امروزی ویندوز از داس مشتق نشده‌اند. در عوض، از یک کرنل جدید که شباهت طراحی بسیاری به VMS دارد استفاده می‌کنند.

در سال 1991، لینوس توروالدز یک دانشجو علوم کامپیوتر در دانشگاه هلسینکی بود. او به مطالعه یونیکس و توانایی‌های کامپیوترهای 86X جدیدی که به تازگی خریداری کرده بود علاقه داشت. لینوس برنامه‌ای که در ادامه به کرنل لینوکس تبدیل شد را به عنوان یک مقلد ترمینال سطح پایین شروع کرد؛ برنامه‌ای برای اتصال به کامپیوترهای بزرگتر دانشگاهش. با رشد برنامه‌اش، او شروع به اضافه کردن امکانات جدید که برنامه ترمینال او را به چیزی که بهتر است به عنوان کرنل لینوکس توصیف شود کرد. سرانجام، او با هدف خلق یک کرنل سازگار با یونیکس، که قابلیت اجرای اکثر نرم افزارهای یونیکس آن زمان را داشت، شروع به نوشتن کرد.

تاریخ یونیکس به دو دهه بیشتر باز می‌گردد؛ به سرچشمه خود در سال 1969. به دلیل اینکه AT&T در آن زمان انحصارا فروشنده تلفن در ایالات متحده بود، قانونا حق فروش نرم افزار را نداشت. در نتیجه، زمانی که کارمندانش یونیکس را خلق کردند، AT&T عملاً سیستم عامل یونیکس را به عموم بخشید. دانشگاه‌ها راجع به انتخاب کردن یونیکس مشتاق بودند و از آنجایی که AT&T سورس کد آن را منتشر کرده بود شروع به تغییر آن کردند. بنابراین، یونیکس دو دهه تاریخچه توسعه نرم افزار اپن سورس برای شروع داشت. به دلیل اینکه یونیکس بر روی طیف گسترده‌ای از سخت افزار‌ها اجرا می‌شد، اکثر برنامه‌های یونیکس به عنوان سورس کد منتشر می‌شدند. برنامه‌های باینری برای یک ماشین به ندرت بر روی یک ماشین متفاوت اجرا می‌شدند.

در اوایل، لینوکس شروع به استفاده از نرم افزارها کرد. همانطور که در فصل 1 ذکر شد، توسعه دهنده‌گان اولیه لینوکس خصوصاً علاقه مند به پروژه نرم افزار غیر یونیکسی گنو (GNU) داشتند؛ در نتیجه لینوکس به سرعت تبدیل به انباسته‌ای از مجموعه ابزارهای گنو شد. بیشتر این نرم افزار توسط ورک استیشن‌ها و کامپیوترهای قوی تر نوشته شده بود؛ اما به دلیل اینکه سخت افزارها به سرعت بهبود می‌یافتند، بر روی کامپیوترهای 86X اوایل دهه 90 میلادی به خوبی اجرا می‌شد.

نکته: سیستم عامل BSD386 یک سیستم عامل رقیب و مشابه یونیکس در اوایل دهه 90 میلادی بود. امروزه، در چندین سیستم عامل مربوطه دیگر شاخه داده است: FreeBSD، NetBSD، OpenBSD، PC-BSD و DragonflyBSD. لینوکس به سرعت توسعه دهنده‌گان علاقه مند دنبال کننده‌ای به دست آورد که پتانسیل آن را در آوردن نرم افزار طبقه ورک استیشن به کامپیوتر دسکتاپ دیدند. این افراد در بهبود

کرنل لینوکس، برای ایجاد تغییر های لازم برای برنامه های یونیکس برای اجرا در لینوکس و نوشتن برنامه های پشتیبانی منحصر به فرد لینوکس تلاش کردند. تا اواسط دهه 90 میلادی، چندین توزیع لینوکس شامل برخی که امروزه هم باقی ماندند ایجاد شدند. (برای مثال، توزیع Slackware در سال 1993 و Redhat در سال 1995 منتشر شدند).



## Real World Scenario

### The Microkernel Debate

لینوکس یک نمونه از کرنل یکپارچه است؛ که به کرنل میگویند که تمام کارهایی که یک کرنل باید انجام دهد را در یک پروسه عظیم انجام میدهد. در سال 1991 یک طرح کرنل رقیب، شناخته شده با نام میکروکرنل، بسیار معروف بود. میکروکرنل ها از کرنل های یکپارچه بسیار کوچکتر هستند؛ آنها تا حد ممکن وظایف را به پروسه های غیر کرنلی منتقل میکنند و سپس ارتباط بین پروسه ها را مدیریت میکنند.

کمی پس از انتشار لینوکس، لینوس توروالدز در یک مناظره عمومی با اندرو تنبام، خالق سیستم عامل مینیکس درگیر شد که توروالدز از آن برای توسعه اولیه لینوکس استفاده میکرد. مینیکس از طراحی میکرو کرنل استفاده میکرد، و تنبام طراحی یکپارچه لینوکس را برعکس محسوب میکرد.

از لحاظ کارآمدی برای کاربر، هر دو طراحی کفایت میکنند. لینوکس و کرنل های BSD از طراحی یکپارچه استفاده میکنند؛ در حالی که نسخه های مدرن ویندوز، GNU HURD و Minix نمونه هایی از میکرو کرنل ها هستند. اگرچه برخی افراد همچنان درگیر این تفاوت ها هستند.

## Seeing Today's Linux World

در اواسط دهه 90 میلادی، مهمترین امکانات لینوکس که امروزه هم وجود دارند ایجاد شد. تغییرات بعد از آن شامل موارد زیر میشود:

**بهبود کرنل:** کرنل لینوکس تغییرات بزرگی از سال 1991 به خود دیده است؛ زمانی که قادر خیلی از امکانات مورد نیاز امروزی ما بود. اصلاحات شامل اضافه شدن امکانات شبکه ای، درایورهای سخت افزاری بیشمار، پشتیبانی امکانات مدیریت انرژی و پشتیبانی از بسیاری از CPU های غیر x86.

**بهبود در ابزارهای پشتیبانی:** موازا با پیشرفت کرنل لینوکس، اصلاحاتی در رابطه با پشتیبانی برنامه هایی که به آنها وابسته است مانند کامپایلرها، GUI، Shells، و غیره انجام داده است.

**خلق ابزارهای پشتیبانی جدید:** ابزارهای پشتیبانی جدید در طی سالها پدیدار شدند و از ابزارهای کوچک تا محیط های بزرگ دسکتاپ متغیر هستند. در واقع، برخی از این ابزارها، مثل محیط دسکتاپ مدرن، بیشتر برای کاربران واضح هستند تا خود کرنل.

**خلق توزیع های جدید:** همانطور که پیشتر اشاره شد، Slackware به سال 1993 و Red Hat (جد بزرگ لینوکس تجاری Red Hat و CentOS و Fedora) به سال 1995 برمیگردند. توزیع های دیگر در بین این سالها ظهور کردند که برخی از آنها بسیار مهم بوده اند. برای مثال سیستم عامل اندروید که بر روی گوشی های هوشمند و تبلت ها استفاده میشود، در دهه های اخیر بسیار موثر بوده است.

ریشه های لینوکس بیشتر در نرم افزارهای اپن سورس دهه های 90 و 80 میلادی باقی مانده است؛ اگرچه کاربر یک دسکتاپ عادی یا یک سیستم عامل تعییه شده معمولاً سیستم عامل را از دید محیط گرافیکی آن درک میکند، بسیاری از چیزهایی که در پشت صحنه رخ میدهد به خاطر کرنل لینوکس و ابزارهای آن که برای چندین دهه است که وجود دارند انجام میشود.

## Using Open-Source Software

فلسفه ای که اکثر توسعه نرم افزاری لینوکس را در بر گرفته با فلسفه ای که توسعه ویندوز از آن مشتق شده متفاوت است. این فلسفه ها نحوه تهیه نرم افزار، کاری که با آن انجام میدهید و چگونگی تغییر آن در طی زمان را مشخص میکند. این بخش این اصول و چگونگی عملکرد لینوکس به عنوان نوعی "آهنربا" که نرم افزار را از چندین منشا در یک جا میگنجاند را توضیح میدهد.

## Understanding Basic Open-Source Principles

به طور کلی، نرم افزار میتواند به چندین شکل متفاوت توصیف شود که هرکدام انتظارات متفاوتی در رابطه با پرداخت، تکثیر و حقوق کاربر دارند. تعداد دسته بندی ها به عمق آنالیز و پیش داوری شخصی که دسته بندی ها را انجام میدهد متغیر است. با این حال چهار دسته برای شروع کافیست:

**نرم افزار تجاری:** اشخاص یا کمپانی ها نرم افزار تجاری را با هدف فروش و سود از آن توسعه میدهند. توسعه دهنگان به طور کلی سورس کد نرم افزار تجاری را محترمانه نگه می دارند که به این معناست که کاربران نمیتوانند تغییراتی در آن ایجاد کنند مگر تغییرات تنظیمات که نرم افزار آن را مقدور میسازد. در گذشته، نرم افزار تجاری در فروشگاه ها و یا با سفارش پستی به فروش میرفت؛ اما امروزه از طریق دانلود اینترنتی فروخته میشود. تکثیر نرم افزار تجاری عموماً غیر قانونی میباشد. مایکروسافت ویندوز و مایکروسافت آفیس هردو نمونه هایی از نرم افزار تجاری هستند.

**اشتراک افزار:** از دیدگاه قانونی، اشتراک افزار به نرم افزار تجاری شباهت دارد؛ از آنجایی که کپیرایت شده است و خالق آن بابتshelf هزینه دریافت میکند. تفاوت بین آنها این است که اشتراک افزار در اینترنت و برخی روش های قدیمی در یک سیستم تعهدی "فروخته" میشود؛ اگر شما از اشتراک افزار فرای زمان مشخص شده استفاده کنید (Free Trial)، موظف میشوید که هزینه آن را پرداخت کنید. اشتراک افزار در سال 1991 رایج بود و امروزه همچنان وجود دارد ولی بسیار کمیاب تر شده است.

**نرم افزار رایگان (Freeware):** این نرم افزارها مانند اشتراک افزارها به صورت رایگان در دسترس هستند؛ با این حال، برخلاف اشتراک افزارها، در تمام مدت استفاده از آن نیازی به پرداخت هزینه نیست. برخی اوقات، نرم افزار رایگان یک نسخه جدا شده از یک اشتراک افزار یا نرم افزار تجاری کامل تر میباشد. در دیگر اوقات،

حالقین آن را به طور رایگان در اختیار میگذارند تا محصولی دیگری را ترویج دهند؛ مثال ها شامل درایور های ویندوز برای بسیاری از سخت افزارها، یا برنامه Adobe Reader برای اجرای فایل های PDF. همچون نرم افزار تجاری و اشتراک افزار، نرم افزار رایگان نیز معمولاً بدون سورس کد منتشر میشود.

**نکته:** نرم افزار رایگان نباید با نرم افزار آزاد که به نرم افزار اوپن سورس مربوط است اشتباه گرفته شود. فصل 2، "Understanding Software Licensing".

**نرم افزار اپن سورس:** نرم افزار اپن سورس توسط 10 اصل توصیف میشود که در OpenSource.org/docs/osd در دسترس است. مهم ترین این اصول عبارت اند از: کاربر حق تکثیر نرم افزار را دارد، سورس کد باید فراهم شود، و کاربر حق ایجاد و تکثیر نسخه های تغییر یافته را دارد. این اصول به این معنا هستند که کاربران میتوانند برنامه های اپن سورس را در راستای نیازشان تغییر دهند حتی اگر در راهی خلاف هدف خالق اصلی باشد.

مغایرتی درون هر یک از این دسته بندی ها وجود دارد و همچنین موارد ترکیب شده ای وجود دارد که در هیچ یک از دسته بندی ها قرار نمی گیرند. برای مثال، سازمان OSI لیستی از لایسنس های تایید شده که با ملاک های آن همخوانی دارد نگه میدارد (OpenSource.org/licenses)؛ با این حال، گاهی توسعه دهندگان نرم افزاری تحت لایسنسی نامفهوم منتشر میکنند و یا لایسنسی استفاده میکنند که قوانین وضع میکند که برخی قوانین نامفهوم تر OSI را زیر پا میگذارند. چنین نرم افزارهایی عملاً اپن سورس محسوب نمی شوند اما نسبت به دیگر بسته بندی ها به دسته اپن سورس نزدیک تر است.

**نکته:** فصل 2 جزئیات بیشتری در رابطه با لایسنس های اپن سورس فراهم میکند.

ایده کلی پشت نرم افزار اپن سورس این است که نرم افزاری که در یک روش شفاف توسعه یافته محتمل است که نسبت به نرم افزاری که در یک روش بسته توسعه یافته برتر باشد. این برتری (و تمام مباحثه علیه آن) در چندین شکل خود را نشان میدهد:

**کد بهتر:** قرار دادن سورس کد در معرض دید عموم به این معناست که میتواند بازبینی شود، قضاؤت شود و توسط هر گروه علاقمندی اصلاح شود. امکان دارد با گهای نامفهومی کشف و نابود شوند چون در غیر این صورت میتوانند باعث ایجاد مشکلاتی در محصول سورس بسته شوند. با این حال، صحت این ادعا توسط مطالعات پشتیبانی نمیشود و پژوهه های کوچکتر ممکن است به نتایج کافی که توجه برنامه نویسان را جلب کند نرسند؛ در نتیجه ممکن است از بازبینی خارجی سورس کد نفع نبرند.

**انعطاف بیشتر:** با فراهم کردن سورس کد برای کاربران، یک پژوهه اپن سورس به کاربر اجازه تغییر نرم افزار در راستای نیازشان را میدهد. اگر کاربران تغییرات را به اشخاص نگهدارنده یا منتشر کننده به عنوان شاخه جدیدی از پژوهه ارائه دهند آن وقت همه میتوانند از این تغییرات استفاده کنند. البته که منتقدان با این موضوع بحث میکنند که این انعطاف پذیری تنها به کسانی که مهارت و زمان لازم برای ایجاد چنین تغییراتی هستند و یا کسانی که پول لازم برای استخدام چنین کسی را دارند نفع می رسانند.

**هزینه کمتر:** اگرچه مفهوم اپن سورس فروش نرم افزار را منع نمی کند، ملزومات تکثیر به این معناست که نرم افزار اپن سورس سرانجام به صورت رایگان در دسترس قرار میگیرد. البته اگر قصد حمایت دارید، می توانید یک قرارداد حمایت خریداری کنید که میتواند سود هزینه ها را کاهش دهد و یا حذف کند.

**عدم قطع دسترسی توسط عرضه کننده:** توسعه دهنگان برخی نرم افزارهای انحصاری، مخصوصاً معروف ترین ها، میتوانند کار را برای محصولات رقیب با استفاده از فایل فرمت ها و استانداردهای انحصاری و یا پشتیبانی نکردن استانداردهای اپن سورس، سخت کنند. ابزارهای اپن سورس کمتر شامل چنین مشکلاتی میشوند از آنجایی که میتوانند به طوری اصلاح شوند که از این استاندارد ها پشتیبانی کنند حتی اگر عملاً این کارو نکنند. اگرچه به عنوان یک موضوع بالفعل، حتی فایل فرمت ها و پروتکل های انحصاری معمولاً مهندسی معکوس میشوند؛ در نتیجه قطع دسترسی توسط عرضه کننده معمولاً یک مشکل موقتی است تا یک مشکل دائمی.

البته که در جامعه لینوکس، اتفاق نظر کلی اجماع این است که این معیارها یک امتیاز واقعی به نفع لینوکس و نرم افزارهای اپن سورس است؛ عیب های گفته شده در مقایسه با مزیت ها عموماً جزئی در نظر گرفته میشوند. در آخر، شما باید خودتان تصمیم خود را در این موضوع ها پس از استفاده کردن انواع مختلف نرم افزار بگیرید.

## Linux as a Software Integrator

یونیکس کمی پس از خلق شدن، به دسته ای از سیستم عامل های ناقص تبدیل شد. این سیستم عامل ها در سطح باینری با یکدیگر ناسازگار بودند ولی در سطح سورس کد کم و بیش ساگاز بودند. امروزه این قضیه همچنان برقرار است. شما میتوانید همان برنامه را برای FreeBSD، سیستم عامل مک و لینوکس تألیف کنید به طوری که بر روی هر سه پلتفرم به یک شکل اجرا شود؛ اما کد های باینری تألیف شده برای یک پلتفرم بر روی دیگری کار نمیکند.

استثناهایی برای این قانون وجود دارد. برخی برنامه ها بر روی امکاناتی که فقط سیستم عامل یونیکس شکل دارند تکیه میکنند. باقی آنها صفاتی دارند که سازگار کردن آنها را برای برخی سیستم عامل ها غیرممکن میکند. اگر یک برنامه رو به زوال برود، ممکن است بر روی سیستم عامل های جدید غیر قابل استفاده شود چون بر کامپایلرها یا امکاناتی از سیستم عامل تکیه میکند که تغییر یافته اند. چنین مشکلاتی در طی زمان رفع می شوند، ولی بصورت دوره ای ناگهان رخ میدهند.

به دلیل شهرت لینوکس، اکثر برنامه های اپن سورس یونیکس با لینوکس به خوبی سازگار هستند. نرم افزار های تجاری نیز برای لینوکس وجود دارند با این وجود اکثر اینها مبهم یا تخصصی هستند. در هر مواقعي، لینوکس سیستم عاملی بوده است که اکثر برنامه های اپن سورس یونیکس باید از آن پشتیبانی کنند. این تاثیر به قدری قوی است که بسیاری از پروژه ها لینوکس را به عنوان پلتفرم اصلی خود انتخاب میکنند.

## Understanding OS Roles

کامپیوترها نقش های بسیاری در جهان دارند، و هم زمان با اینکه کامپیوترها بسیار رایج تر و ارزان تر شده اند، این نقش ها چند برابر می شوند. لینوکس میتواند در اکثر این نقش ها به عنوان سیستم عامل خدمت کند که هر کدام زیر مجموعه ای ابزارهای پشتیبانی خود را دارد. برخی از این نقش ها نیاز به تغییرات خود کرنل را دارند. ما به طور مختصر سه عدد از این نقش ها را توضیح میدهیم: کامپیوترهای تعبیه شده، کامپیوترهای دسکتاپ و لپ تاپ ها، و کامپیوتر سرورها.

## Looking at Embedded Computers

همانطور که در فصل 1 ذکر شد، کامپیوترهای تعبیه شده دستگاه های تخصصی ای هستند که هدفی خاص را انجام میدهند. مثال ها شامل زیر می شود:

**گوشی موبایل:** گوشی های هوشمند مدرن از کامپیوترهایی با سیستم عامل استفاده میکنند که از ساده تا پیچیده متغیر هستند. لینوکس به بعضی از این گوشی های موبایل نیرو میبخشد؛ معمولاً در شکل سیستم عامل اندروید.

نکته: اپل، مایکروسافت و دیگر عرضه کنندگان سیستم عامل های خود را برای گوشی های موبایل فراهم میکنند.

**کتابخوان مجازی:** این دستگاه ها، همانند گوشی موبایل، کامپیوترهای تخصصی ای هستند که از یک سیستم عامل برای عملکرد استفاده میکنند. سیستم عامل بسیاری از کتابخوان های مجازی، لینوکس است که یا یک نسخه شخصی سازی شده است یا اندروید.

**مانیتور های IoT:** اینترنت اشیا (IoT) از سیستم های تعبیه شده کوچکی تشکیل شده است که شرایط فیزیکی مثل دما، رطوبت، روشنایی یا حرکت را تحت نظر میگیرد و از آن اطلاعات برای کنترل موتورها، سوییچ ها، دوربین ها و دیگر دستگاه ها استفاده میکند.

**DVR:** ویدیو رکوردر دیجیتال (DVR)، که برنامه های تلویزیونی را برای تماشای دوباره ضبط می کند، کامپیوترهایی با نرم افزار تخصصی هستند. برخی از اینها، از جمله مدل های معروف TiVo، از لینوکس استفاده میکنند.

نکته: پکیج MythTV، میتواند یک کامپیوتر PC عادی را به یک DVR لینوکسی تبدیل کند، با این حال شما نیاز به یک TV Tuner و دیگر سخت افزارهای مخصوص برای انجام این کار دارید. (MythTV.com)

**کامپیوترهای صنعت خودرو:** اتومبیل ها سال ها شامل کامپیوترها میشدند. اینها اکثرا تعبیه شده اند تا بر روی موتور و دیگر سیستم های خودرو نظارت و کنترل داشته باشند؛ با این حال، خودرو های مدرن به طور فزاینده با کامپیوترهایی عرضه می شوند که کاربر به سهولت آنها را به عنوان کامپیوتر تشخیص دهد. آنها

GPS و سیستم جلوگیری از تصادف را مدیریت میکنند، ترمز های اضطراری را تنظیم میکنند، سیستم صوتی را کنترل میکنند و حتی دسترسی به اینترنت را مهیا میکنند.

**اسباب و لوازم:** تلویزیون ها، یخچال ها و دیگر اسباب و لوازم به طور فزاینده از کامپیوترها برای دانلود آپدیت نرم افزار، نظارت بر استفاده انرژی و دیگر اهداف استفاده میکنند.

ممکن است فکر کنید که کامپیوترهای تبلت نیز درحال جا گرفتن در این دسته بندی هستند، با اینکه می توانند شباهت های بیشتری به کامپیوترهای دسکتاپ و لپ تاپ داشته باشند. تفاوت به طور کلی در این است که کاربر چقدر کنترل بر روی سیستم عامل دارد؛ دستگاه های تعییه شده توسط کاربران مصرف کننده نگهداری نمیشنوند و صرفا استفاده میشنوند. امور مدیریتی سیستم توصیف شده در این کتاب از قبل در کارخانه و یا توسط یوزر اینترفیس های تخصصی تر و ساده تر انجام شده اند.

## Exploring Desktop and Laptop Computers

لینوکس حیات خود را بر روی یک کامپیوتر دسکتاپ آغاز کرد و با اینکه لینوکس حتی نزدیک به فتح بازار آن نیست اما کامپیوترهای دسکتاپ راه خوبی برای آغاز یادگیری لینوکس می باشد. از لحاظ دیدگاه مدیریت سیستم، لپ تاپ ها به کامپیوترهای دسکتاپ شباهت دارند؛ هر دو آنها معمولاً توسط تعداد کمی از افراد برای کارهایی مثل پردازش کلمات، جستجوی وب و مدیریت عکسهای دیجیتال استفاده می شوند. برای اختصار، از دسکتاپ برای اشاره به هردو مدل استفاده میکنیم.

**نکته:** کامپیوترهای دسکتاپ به گروهی دیگر از کامپیوترها شباهت دارند که با نام *Workstation* شناخته میشنوند. ورک استیشن ها عموماً قوی تر و تخصصی تر هستند و معمولاً از لینوکس یا یونیکس استفاده میکنند

برای چنین استفاده هایی نرم افزارهای لینوکسی به طور گسترده وجود دارند و از کیفیت بالایی برخوردارند؛ با این حال بعضی افراد، همتا های تجاری مانند مایکروسافت آفیس یا فتوشاپ را ترجیح میدهند که برای لینوکس وجود ندارند. این مزیت برای یک سری نرم افزارهای تجاری خاص دلیل فاتح بودن مایکروسافت در بازار دسکتاپ است. برخی افراد معتقدند که مدل توسعه اپن سурс به خود زحمت ساخت نرم افزار GUI محبوب را نمی دهد چون توسعه دهنگان نرم افزار معمولاً برای درک کامل نیازهای کاربران بیش از حد متمایل به امور فنی هستند. بدون یک راه واضح برای ملزم کردن توسعه دهنگان برای تکمیل این نیازها، که نرم افزارهای تجاری آنها را ایجاد میکنند، پروژه های اپن سурс مقداری از رقبای تجاری خود عقب خواهند افتاد. نرم افزارهای خاصی که برای تمام کامپیوترهای دسکتاپ لینوکسی لازم هستند شامل موارد زیر میشود:

- رابط گرافیکی X Window System (به اختصار X)
- یک محیط محبوب دسکتاپ مانند Xfce، GNOME، KDE، Unity یا Mozilla Firefox
- یک مرورگر وب، مثل Evolution یا Thunderbird
- یک Email Client، مانند

- یک ادیتور گرافیکی، مثل GIMP
- یک مجموعه آفیس، مانند OpenOffice یا مشابه آن

الزامات اضافی بر اساس نیازهای کاربر متغیر است؛ برای مثال، یک کاربر ممکن است به ابزارهای ادیت مالتی مدیا نیاز داشته باشد، در حالی که دیگری ممکن است به نرم افزار تحلیل داده علمی نیاز داشته باشد.

توزیع های لینوکس مثل Ubuntu و Fedora معمولاً به صورت دیفالت این ابزارها را در خود دارند، یا به عنوان یک گروه از طریق گزینه Single Install-Time نصب میشوند. این توزیع ها برای نگهداری نسبتاً راحت طراحی شده اند تا کاربران با مهارت های نسبتاً کمی قادر به نصب و اجرای سیستم عامل در طی زمان باشند.

## Investigating Server Computers

کامپیوترهای سرور از لحاظ سخت افزاری می توانند تقریباً با کامپیوترهای دسکتاپ یکی باشند؛ اگرچه سرورها گاهی اوقات بسته به استفاده آنها، نیاز به هارد درایو بزرگتر یا اتصال شبکه بهتری دارند. بسیاری از برنامه های شبکه سرور اولین بار برای یونیکس یا لینوکس نوشته شده اند که باعث میشود این پلتفرم ها بهترین گزینه برای اجرای آنها باشند. مثال ها شامل زیر میشود:

- وب سرورها، مثل Apache
- ایمیل سرورها، مانند Sendmail و Postfix
- دیتابیس ها، مثل MySQL
- فایل سرورها، مانند Samba یا NFS (Network File System)
- پرینت سرورها، مثل Samba (Common Unix Printing System) یا CUPS (Berkeley Internet Name Domain)
- سرورهای DNS، مانند BIND (Internet Software Consortium's)
- سرورهای DHCP، مانند ISC's (Network Time Protocol)
- تایم سرورها، مانند NTP (Secure Shell) یا VNC
- سرورهای ریموت لایکن، مثل SSH یا VNC (Secure Shell)

نکته: سرورهای ریموت لایکن به کاربران این قابلیت را میدهد که برنامه های دسکتاپ مانند را به صورت ریموت بر روی کامپیوتر اجرا کنند. درنتیجه، گاهی اوقات حتی بر روی سیستم های دسکتاپ نیز یافت میشوند.

در سازمان های بزرگ، هرکدام از این سرویس ها ممکن است یک کامپیوتر سرور کاملاً جدا داشته باشد. اگرچه این امکان وجود دارد که یک کامپیوتر بسیاری از این برنامه های سرور را به صورت همزمان اجرا کند. بسیاری از این سرورها نیازی به رابط گرافیکی ندارند؛ درنتیجه کامپیوترهای سرور بدون X، محیط دسکتاپ یا برنامه های عمومی دسکتاپ که بر روی یک کامپیوتر دسکتاپ پیدا میکنید همچنان قادر به انجام عملیات هستند. یکی از مزیت های لینوکس در برابر ویندوز این است که شما میتوانید کامپیوتر را بدون این عناصر به کار بیندازید، و حتی می توانید آنها را کاملاً حذف کنید؛ این به این معناست که GUI به طور غیر لازم منابع سیستم مانند RAM را مصرف نخواهد کرد. علاوه بر این، اگر یک آیتم مانند X درحال اجرا نیست، هرگونه

باگ امنیتی که ممکن است ایجاد کند بی اهمیت خواهد بود. برخی توزیع ها، مانند Arch، Debian و Gentoo از پیکربندی ابزارهای GUI اجتناب میکنند. این باعث میشود که این توزیع ها برای کاربران جدید ناآشنای شود، ولی تکیه بر ابزارهای پیکربندی متنی برای ادمین های باتجربه کامپیوترهای سرور مشکلی نخواهد بود.

این افراد که از کامپیوترهای سرور زیادی نگهداری می کنند عموماً از لحاظ فنی بسیار خبره هستند و گاهی در پروژه های سرور اپن سورسی که خودشان استفاده میکنند کمک میکنند. این همکاری نزدیک بین کاربران و برنامه نویسان می تواند در کمک به بروز نگه داشتن اپلیکیشن ها و ابزارهای لازم در دنیای واقعی کمک کند.

به خاطر داشته باشید که تفاوت بین دسکتاپ و کامپیوتر سرور مطلق نیستند؛ یک کامپیوتر می تواند ترکیبی از هر دو نوع نرم افزار را اجرا کند. برای مثال، شما میتوانید کامپیوترهای دسکتاپ را در محیط اداری به طوری تنظیم کنید که سرور فایل سیستم را اجرا کند. در یک خانه یا یک دفتر کار کوچک، راه اندازی سرورهای دیگر بر روی کامپیوتر دسکتاپ میتواند نیاز خرید سخت افزارهای تخصصی برای پر کردن آن نقش ها را رفع کند.

# Chapter 4

## Using Common Linux Programs

این فصل برخلاف فصل های قبل که اطلاعاتی خشک وغیرعملی داشتند، نگاهی عملی تر به لینوکس دارد. این فصل با نگاهی بر فضای دسکتاپ لینوکس آغاز میشود که شامل اطلاعاتی درباره رایج ترین محیط های دسکتاپ و کاربردهای معمول آن میشود. اگر شما از یک محیط دسکتاپ استفاده می کنید، به احتمال زیاد برای این است که نرم افزارهای کاربردی را اجرا کنید؛ در این فصل، شما راجع به چند پکیج کاربردی رایج لینوکس خواهید آموخت. به علاوه، ممکن است مایل باشید که نرم افزارهای کاربردی اضافی هم نصب کنید، پس مدیریت پکیج نرم افزار در آخر فصل به طور مختصر توضیح داده شده است.

یک استفاده جامع دیگر از یک سیستم لینوکس، سرور شبکه است؛ پس این فصل چند برنامه رایج سرور که ممکن است با آنها برخورد داشته باشید را دربر میگیرد. با اینکه ممکن است شما نیازی به نوشتن برنامه نداشته باشید، ولی با اینحال ممکن است مجبور به کامپایل کردن برنامه از سورس کد باشید؛ پس باید با ابزار برنامه نویسی رایج لینوکس که در این فصل توصیف می شوند آشنا باشید.

## Using a Linux Desktop Environment

احتمال اینکه اولین تجربه شما با یک سیستم عامل لینوکس شامل یک محیط دسکتاپ شود بالا میباشد. محیط دسکتاپ مجموعه ای از برنامه ها است که اسکرین را کنترل میکنند و همچنین برنامه های ابزاری کوچکی را برای اجرای امور مدیریت فایل ها فراهم میکند. لینوکس چندین محیط دسکتاپ را تدارک دیده است تا اگر یکی از آنها مورد پسند واقع نشد، بتوانید یکی دیگر را انتخاب کنید. علاوه بر ارائه اطلاعات درباره محیط های دسکتاپ قابل دسترس، این بخش چند ابزار مورد استفاده برای اجرای برنامه ها و مدیریت فایل ها را معرفی میکند.

## Choosing a Desktop Environment

بسته به توزیع لینوکس و تنظیمات نصبی شما، احتمال اینکه سیستم شما دارای بیش از یک محیط دسکتاپ باشد بالاست. رایج ترین محیط های دسکتاپی شامل موارد زیر هستند:

**KDE Plasma**: محیط دسکتاپ K پلاسما (KDE) یک محیط دسکتاپ معروف برای لینوکس، و دیفالت برای openSUSE می باشد و دارای ابزارهای قدرتمندی است و با استفاده از QT Widget Set ساخته شده است .(kde.org)

**نکته:** یک کتابخانه است که امکانات GUI مثل منوها و دیالوگ باکس ها را مدیریت میکند. QT و GTK+ (بخشی از پروژه گنو) دو مورد از سنت های ویجت در لینوکس های امروزی هستند.

**GNOME**: گنوم (gnome.org) یکی دیگر از محیط های دسکتاپ معروف در لینوکس است که محیط دیفالت توزیع های فدورا و اوبونتو است. گنوم بر روی (+) GTK ساخته شده است و همانند KDE Plasma، گنوم شامل بسیاری از ابزارهای قدرتمندی است که با یکدیگر در تعامل اند. هدف گنوم فراهم کردن یک محیط دسکتاپ آسان برای استفاده است.

**Cinnamon**: این محیط دسکتاپ در اصل بر پایه گنوم است و در ابتدا فقط برای توزیع لینوکس مینت منتشر شد اما امروزه توسط باقی توزیع ها نیز پشتیبانی میشود. در آسانی استفاده، انعطاف پذیری، ظاهر تمیز و تجربه مساعد کلی، Cinnamon یک محیط دسکتاپ عالی برای کسانی که تازه با لینوکس آشنا شده اند میباشد اگرچه کاربران با تجربه لینوکس نیز آن را ترجیح میدهند.

**LXDE** : محیط دسکتاپ سبک وزن X11، یا به اختصار LXDE، همانطور که از نامش پیداست، هدفش مصرف منابع کمتر و در نتیجه عملکرد بهتر بر روی کامپیوتر های قدیمی و نسبتاً ضعیف است. این محیط دسکتاپ نیز بر روی GTK+ ساخته شده است. LXDE معمولاً محیط دسکتاپ دیفالت بر روی توزیع هایی است که اصلی ترین هدف‌شان مصرف منابع کمتر نا حد ممکن در کنار عملکرد بی نقص باشد (lxde.org).

**Xfce**: این محیط دسکتاپ سبک معروف در xfce.org قابل دسترس است. در ابتدا از روی یک محیط دسکتاپ تجاری به نام CDE مدل سازی شده بود، اما توسط GTK+ ساخته شده است. Xfce قابلیت پیکربانی بیشتری نسبت به گنوم فراهم کرده است. این محیط دسکتاپ اجرا و Load سریعی دارد اما نسبت به اکثر نمونه های دیگر منابع کمتری مصرف میکند.

**Build Your Own**: این امکان وجود دارد که محیط دسکتاپ خاص خودتان را با اجزای مورد نظر خود بسازید. از آنجایی که این امر می تواند پیچیده باشد، بهترین راه این است که با یک راهنمای پر جزئیات شروع کنید؛ موتور جستجوی مورد علاقه خود را باز کنید و عبارت *How to create your own linux desktop environment* را تایپ کنید تا اطلاعات دقیقی در رابطه با ساخت دسکتاپ کاستوم پیدا کنید. در کمترین حالت، شما نیاز به یک ویندو منیجر دارید. با این حال، برای اینکه پیکربندی ها یک محیط دسکتاپ واقعی باشند، شما به اجزای دیگری مانند فایل منیجر و ابزارهای کاربردی کوچیک نیاز خواهید داشت. تمام اجزا باید توسط نوعی سیستم منو قابل دسترس باشند.

متاسفانه، این امکان وجود ندارد که راهنمایی های دقیقی درباره اینکه چه زمانی یک محیط دسکتاپ از دیگری بهتر کار میکند فراهم کرد. با این حال، پیشنهاداتی که در ادامه ذکر میشود میتواند کمک کننده باشد. کاربران جدیدی که به ویندوز و مک عادت دارند احتمالاً از KDE Plasma بیشترین رضایت را خواهند داشت. KDE Plasma به این سیستم عامل های دسکتاپ سنتی شباهت بیشتری دارد. هدف گنوم در لطفت و آسانی است، در نتیجه برای کسانی که خواهان یک یوزر اینترفیس خوش ظاهر هستند مناسب است. Xfce و LXDE نیز گزینه های خوبی برای سیستم هایی که رم پایین یا CPU های ضعیفی دارند هستند. کسانی که مایل اند همه چیز را کاستوم سازی کنند یا کامپیوترهای ضعیف تری دارند بهتر است بر روی روش Build Your Own بیشتر تحقیق کنند.

قبل از اینکه تصمیم بگیرید به یک محیط دسکتاپ بخصوص بچسبید، بهتر است دو یا سه مورد از آنها را امتحان کنید. در اکثر مواقع، شما میتوانید چندین محیط دسکتاپ را با استفاده از پکیج منیجر، که در این فصل و با جزئیات بیشتر در فصل نهم توصیف می‌شود نصب کنید.

بعد از آن که محیط‌های دسکتاپ اضافی شما نصب شدند، شما میتوانید زمانی که در کامپیوتر خود با استفاده از یک منو لایکن می‌کنید آنها را انتخاب کنید. مثال در شکل 4.1 یک لایکن اسکرین در لینوکس مینت را نشان میدهد.

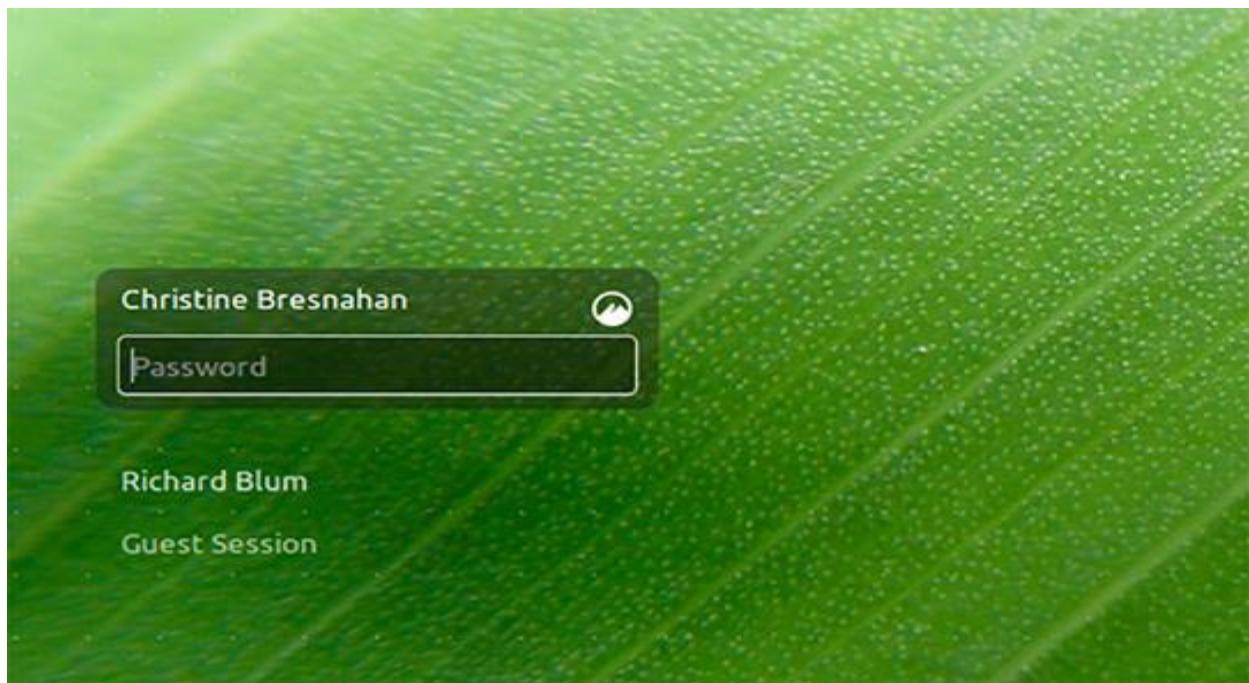


Figure 4.1 A typical Linux desktop login screen where you choose the user account

برای انتخاب محیط دسکتاپ در لینوکس مینت، باید برو روی گزینه‌ای با نماد قله کوه در کنار یوزرنیم کلیک کنید. در شکل 4.2 مثالی از نتیجه این کار نشان داده شده است.



Figure 4.2 GUI login managers usually provide a selection of desktop environments from which you can choose.

گزینه های منو بر روی سیستم لینوکس شما بسته به محیط های دسکتاپ دیفالت و نصبی متغیر خواهد بود. چگونگی انتخاب محیط دسکتاپ نیز در هر توزیع متغیر است؛ پس شما باید گزینه های لاگین اسکرین خود را دنبال کنید تا محیط دسکتاپی مورد نظرتان را انتخاب کنید.

## Launching Programs

اکثر محیط های دسکتاپی راه های مختلفی برای اجرای برنامه ها ارائه میکنند. جزئیات میتواند در هر محیط نسبت به دیگری به طور قابل توجهی متفاوت باشد. با این حال، مثال های مفید شامل موارد زیر میشود:

**منوهای دسکتاپ:** اکثر محیط های دسکتاپ منوهایی را در بالا، پایین و اطراف گوشه اسکرین فراهم میکنند. یک یا بیشتر از یک آیتم در این منوها میتواند به شما دسترسی به مجموعه ای از اپلیکیشن های از پیش تعیین شده را بدهد.

**آیکون های دسکتاپ:** در برخی محیط های دسکتاپ شما میتوانید آیکون ها را در محدوده اصلی دسکتاپ قرار دهید. با کلیک یا دابل کلیک بر روی این آیکون ها برنامه را اجرا میکند. برخی پیکربندی های دیفالت بعضی اپلیکیشن ها را در محدوده اصلی دسکتاپ قرار میدهد.

**پنل ها:** بعضی محیط های دسکتاپ پنل هایی را فراهم میکند که آیکون اپلیکیشن های رایج در آن ظاهر میشود؛ معمولاً پنل ها در اطراف اسکرین قرار دارند. GNOME Shell (یک ورژن از محیط دسکتاپ گنوم) از چنین پیکربندی ای بصورت دیفالت استفاده میکند-با اینکه پنل تنها زمانی ظاهر میشود که شما بر روی آیتم در گوشه بالا-چپ اسکرین کلیک کنید.

**Context Menus**: گاهی اوقات شما میتوانید با کلیک راست بر روی یک بخش استفاده نشده یک Context Menu با انواعی از آپشن ها، که می تواند شامل آپشنی برای اجرای برنامه ها باشد، بسازید.

**جستجوی برنامه ها**: برخی محیط های دسکتاپ یک قابلیت جستجو فراهم میکنند که شما میتوانید از آن برای پیدا کردن یک برنامه با نام آن استفاده کنید. این قابلیت جستجو میتواند بر روی اسکرین اصلی یا داخل منوهای دسکتاپ باشد. معمولاً، شما بخشی از نام برنامه را تایپ میکنید و برنامه هایی که نام آنها تطابق دارند در لیست ظاهر می شوند؛ سپس شما میتوانید برنامه مدنظر را از لیست انتخاب و اجرا کنید.

**ترمینال ها**: شما میتوانید برنامه ای با نام ترمینال را که یک یوزر اینترفیس متنی را درون یک پنجره فراهم میکند را اجرا کنید؛ سپس میتوانید با تایپ کردن نام فایلی برنامه های متنی یا گرافیکی، آنها را اجرا کنید. این شیوه در فصل 5 با جزئیات بیشتری بررسی میشود.

برای اینکه بهتر این روش ها را درک کنید، چند مثال مفید خواهند بود.  
در ابتداء، شما مرورگر فایرفاکس را در توزیع فدورا 30 ورک استیشن با استفاده از محیط دسکتاپ گنوم اجرا میکنید.

گام های زیر را پس از آنکه در سیستم خود لاگین کردید دنبال کنید:

1. بر روی آیتم Activities در گوشه بالا-چپ اسکرین کلیک کنید. نتیجه یک پنل خواهد بود (با نام Favorites) در سمت چپ اسکرین، همانطور که در شکل 4.3 نشان داده شده است.
2. موس را بر روی آیکون فایرفاکس ببرید که آیکون اول در شکل 4.3 می باشد.
3. بر روی آیکون فایرفاکس کلیک کنید. پس از مدت کوتاهی، یک پنجره فایرفاکس باید باز شود.

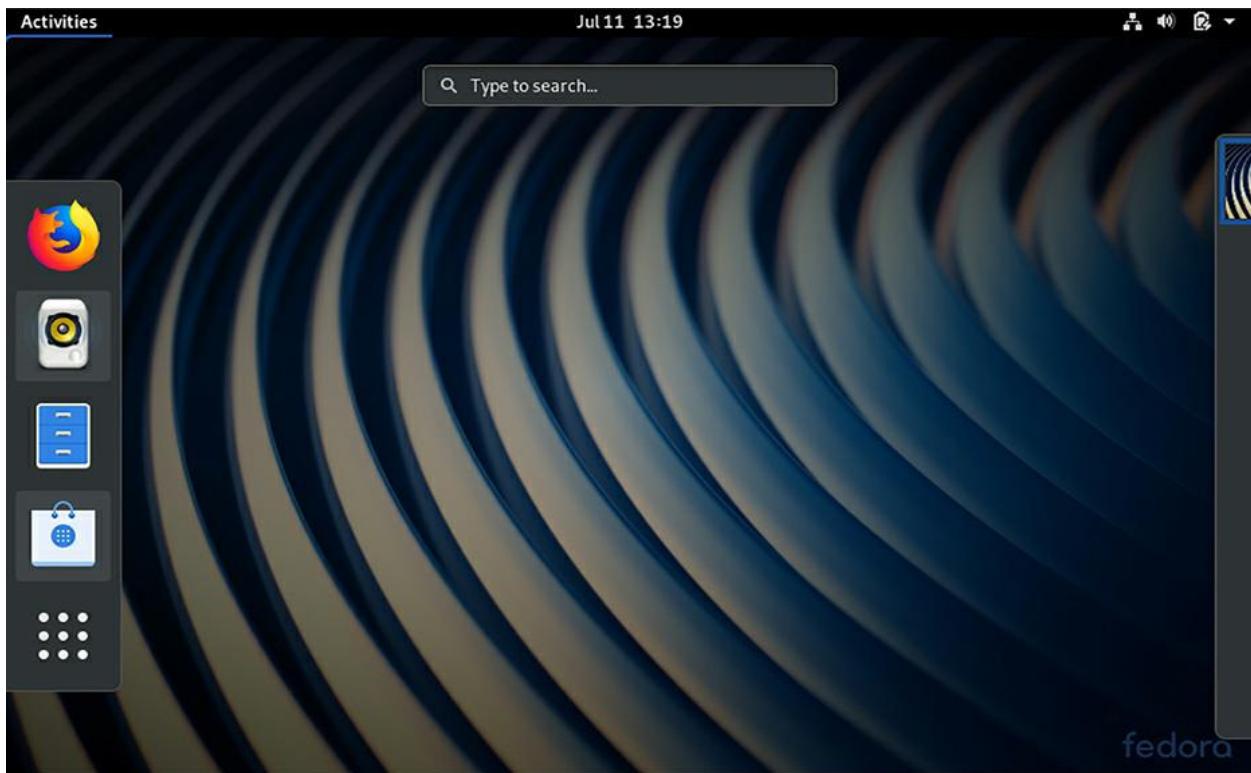


Figure 4.3 With panels you can launch popular programs in GNOME and some other desktop environments.

راه های دیگری برای این کار وجود دارد، مانند تایپ کردن نام برنامه در بخش جستجو (قابل مشاهده در بخش بالا-وسط در شکل 4.3). از آنجایی که فقط تعداد محدودی از برنامه ها در پنل گنوم شل ظاهر میشوند، شما یا باید برنامه ها را به آن اضافه کنید یا باید برنامه هایی که توسعه دهنگان فدورا به طور دیفالت شامل نشدند را به روشی دیگر اجرا کنید.

برای مقایسه، Cinnamon تحت لینوکس مینت 18.3 چند راه واضح برای اجرای فایرفاکس ارائه میکند:

- با کلیک بر روی آیکون آن در سمت چپ پنل پایین اسکرین (شکل 4.4 را مشاهده کنید).
- با پیدا کردن آیکون آن در پنل Favorites. با کلیک بر روی آیکون منو در سمت چپ پنل پایینی اسکرین، این پنل را خواهید دید. پنل Favorites در چپ پنجره منو قرار دارد (شکل 4.4 را مشاهده کنید).
- با پیدا کردن آن در لیست داخل سیستم منو، فایرفاکس را با ورود به سیستم منو، انتخاب کردن اینترنت و کلیک بر روی آیکون فایرفاکس اجرا میکنید.

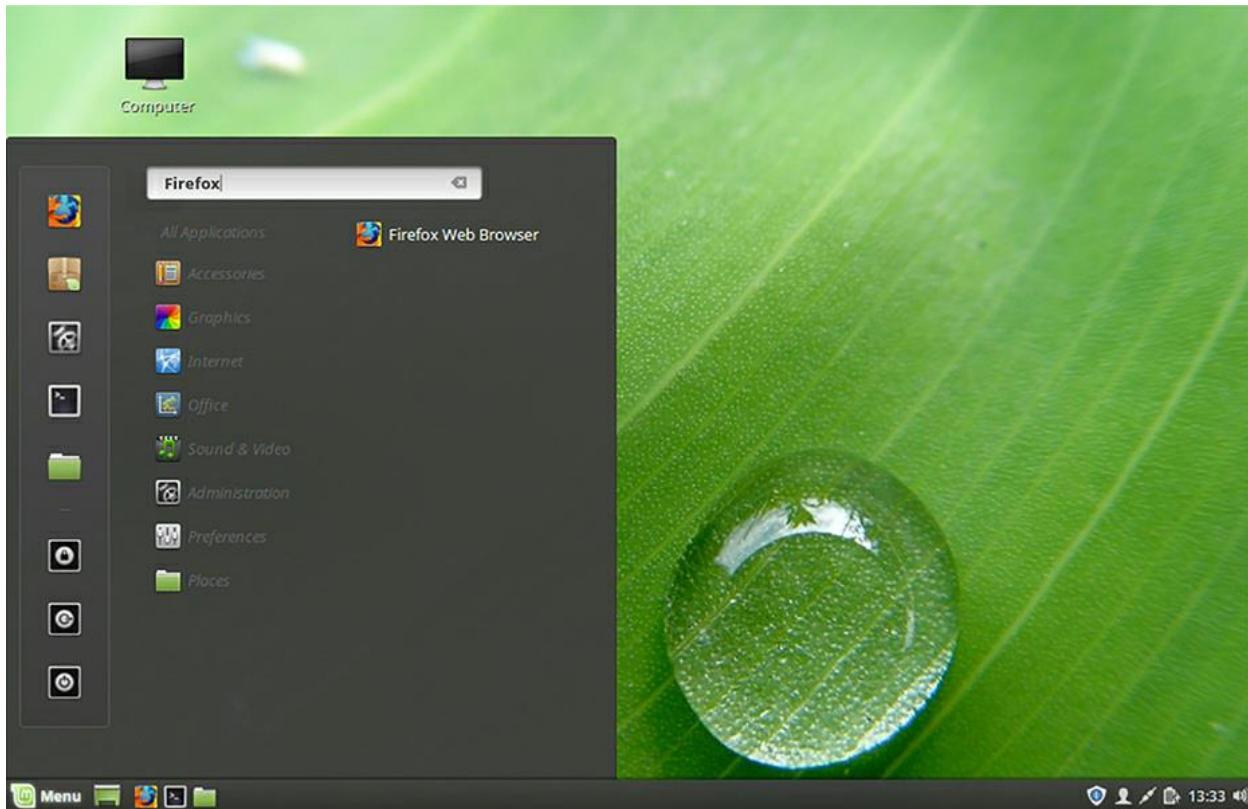


Figure 4.4 Cinnamon's desktop interface provides launch methods similar to those available in Windows.

با محیط های دسکتاپ متنوع، گستره‌های ترین آپشن های اجرا برای تعداد محدودی از اپلیکیشن های محبوب مثل فایرفاکس در دسترس است. برای برنامه هایی که از محبوبیت کمتری برخوردارند ممکن است نیاز به استفاده از روش های نسبتاً پیچیده تری داشته باشید، مثل پیدا کردن برنامه در لیست اپلیکیشن ها. با این حال، شما میتوانید با پیکربندی محیط دسکتاپ برنامه های مدنظر خود را اضافه کنید.

نکته: هر توزیع، دیفالتس های خود را به روش خود تنظیم میکند. پیکربندی گنوم یا Cinnamon مورد استفاده شما ممکن است با تصاویر نشان داده شده در اینجا برابری نکند.

## Using a file manager

اگر به ویندوز و مک عادت دارید، پس مطمئناً از یک فایل منیجر برای مدیریت فایل های خود استفاده کرده اید. لینوکس نیز برای چنین هدفی یک فایل منیجر فراهم کرده است؛ در واقع، شما چندین انتخاب دارید اگرچه اکثرشان به یک طریق عمل میکنند. به عنوان مثال، GNOME Files (Nautilus) را در نظر بگیرید (سابق)، که فایل منیجر پیش‌فرض گنوم است. اگر شما از GNOME Shell روی فدورا استفاده کرده باشید، گنوم فایلز (که گاهی صرفاً فایلز نام دارد) ظاهری شبیه به کمد فایل ها در پنل Favorites دارد، همانطور که

در شکل 4.3 مشخص است. محیط دسکتاپ شما ممکن است یک فایل منیجر را زمانی که یک USB فلش یا یک DVD به دستگاه وارد میشود اجرا کند. شکل 4.5 گنوم فایلز را در لحظات پس از نصب نشان میدهد.

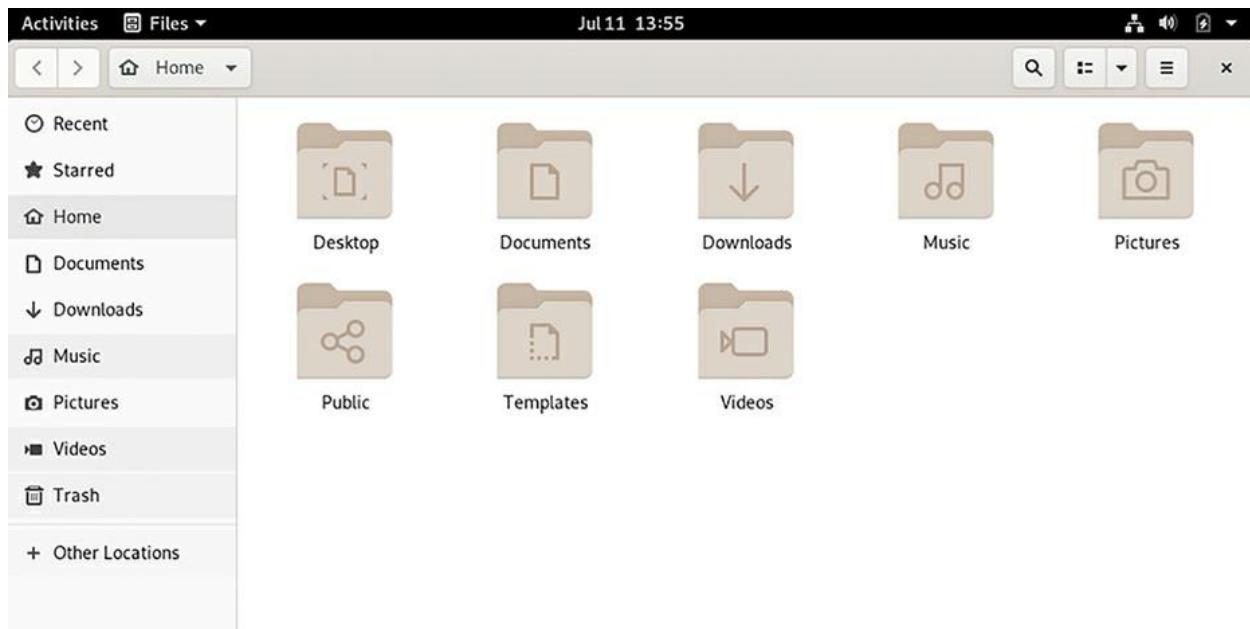


Figure 4.5 GNOME Files provides a view of your files similar to that in other OSs' file managers.

نکته: به غیر از گنوم فایلز، فایل منیجر های دیگری مانند Nemo (فایل منیجر Cinnamon)، Thunar (فایل منیجر KDE Plasma) و Dolphin (Xfce) هم وجود دارند.

از آنجایی که گنوم فایلز به فایل منیجرهای سیستم عامل های دیگر مشابه است، این احتمال وجود دارد که شما به راحتی بتوانید از آن استفاده کنید. برخی از آیتم ها که نیاز به اشاره دارند:

**Home**: این آیتم به دایرکتوری اصلی اشاره دارد که محل ذخیره فایل های شماست. معمولاً، شما تمام فایل های خود را در دایرکتوری اصلی میسازید. صفحه دیفالت گنوم فایلز زمانی که آن را اجرا میکنید، دایرکتوری اصلی است، همانطور که در شکل 4.5 نشان داده شده است. پنل سمت راست فایل ها و دایرکتوری های زیرمجموعه را نشان میدهد.

**Starred**: شما میتوانید بوک مارک اضافه کنید (در گنوم فایلز Starred نام دارد) برای محل هایی که در پنل اصلی نشان داده نمی شوند. به فolder بالاتر محل مورد نظر بروید و با راست کلیک بر روی آیکون فایل، یک منوی کشویی باز کنید (در شکل 4.6 نمایش داده شده است). برو روی گزینه Star در لیست منو کلیک کنید. فایل های جدیدا اضافه شده در بوک مارک ها در گنوم فایلز در محل پنل Starred ظاهر میشوند. شما میتوانید فایل ها را نیز بوک مارک کنید.

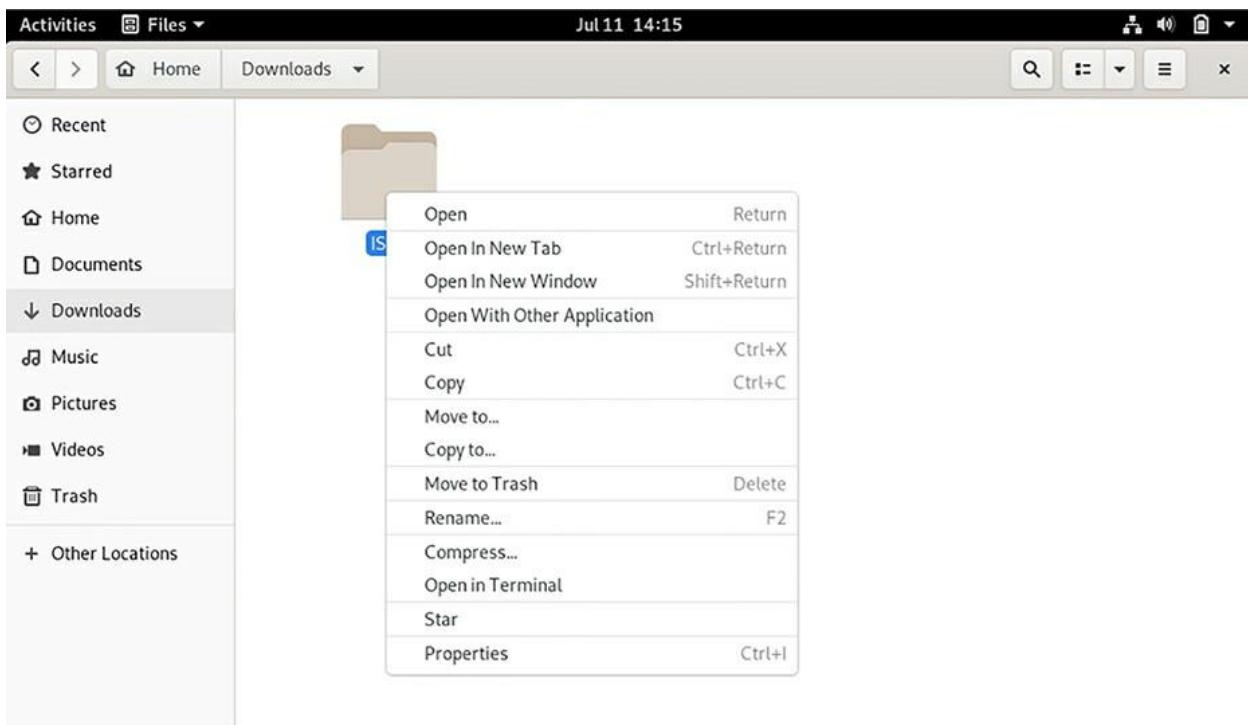


Figure 4.6 You can star folders to enable quick access to directories that interest you.

نکته: اگر بر روی یک محل دابل کلیک کنید، گنوم فایلز سعی بر دسترسی به آن میکند.

شما میتوانید بر روی یک فایل راست کلیک کنید و گزینه **Properties** را در منوی کشویی انتخاب کنید. این باعث ایجاد یک دیالوگ باکس Properties میشود، همانطور که در شکل 4.7 نمایش داده شده است. بخش Open With به شما امکان مرتبط کردن یک نوع فایل به اپلیکیشن ها را میدهد.



Figure 4.7 GNOME Files lets you associate document types with applications.

## Working on Productivity Software

محدوده نرم افزارهای کاربردی بسیار گستردگی دارد. صدها، اگر نه هزاران نرم افزار کاربردی وجود دارد و کتابهایی تماماً بر اساس بسیاری از آنها نوشته شده است. در نتیجه، در این فصل ما اسم‌ها و توضیحات مختصراً از برخی از نرم افزارهای کاربردی رایج برای موضوعات رایج را ارائه می‌دهیم. موضوعات رایج ابزارها شامل مرورگرهای وب، ایمیل کلاینت‌ها، ابزارهای آفیس، نرم افزارهای مولتی مدیا، رایانش ابری و نرم افزارهای موبایل می‌باشد. پیش از توصیف این ابزارها، بهتر است چند نکته راجع به چگونگی پیدا کردن نرم افزارهایی برای انجام کارهای بخصوص در لینوکس را بررسی کنیم.

## Finding the Right Tool for the Job

لینوکس نرم افزارهای کاربردی بسیاری را در موضوعات گسترده‌ای فراهم می‌کند؛ اما اگر تا الان با این زمینه آشنایی نداشته اید ممکن است پیدا کردن آنها کمی دشوار باشد. این مخصوصاً حقیقت دارد از آنجایی که نام نرم افزارها همیشه هدف آنها را مشخص نمی‌کند.

چند تکنیک برای کمک کردن در پیدا کردن نرم افزار مناسب:

**استفاده از منوهای دسکتاپ:** شما میتوانید با استفاده از منوها یا دیگر اپلیکیشن‌های نمایشگر ابزار در دسکتاپ خود، نرم افزارهای کاربردی را پیدا کنید. چنین ابزارهایی معمولاً نرم افزارها را به روش‌های مفیدی دسته بندی می‌کنند. برای مثال، لانچر اپلیکیشن در KDE (در شکل 4.4 نمایش داده شده) نرم افزارها را به کتکتوری‌ها (اکسسوری، گرافیک، اینترنت و غیره) و زیر مجموعه‌های آن، مثل عکاسی و اسکن در کتکتوری گرافیک دسته بندی می‌کند. این میتواند در پیدا کردن نرم افزار مفید باشد، اما فقط در صورتی که نرم افزار نصب شده باشد.

**استفاده از ابزار سرچ:** شما میتوانید از ابزار سرچ در محیط دسکتاپ یا مرورگر وب برای پیدا کردن نرم افزار مناسب استفاده کنید. تایپ یک کلمه یا عبارت کلیدی، مثل **Office** (همراه با کلمه **Linux** اگر از مرورگر استفاده می‌کنید) میتواند به شما در پیدا کردن نرم افزارهای آفیس (پردازشگرهای کلمه، Spreadsheet و غیره) کمک کند.

**استفاده از جدول مشابهات:** اگر شما معمولاً از نرم افزارهای مخصوص ویندوز استفاده می‌کردید، می‌توانید جایگزین‌های لینوکسی آنها را در جدول مشابهات مثل نمونه زیر پیدا کنید:  
[Wiki.linuxquestion.org/wiki/linux\\_software\\_equivalent\\_to\\_windows\\_soft](http://Wiki.linuxquestion.org/wiki/linux_software_equivalent_to_windows_soft)

**استفاده از تجربیات دیگران:** شما میتوانید با سوال کردن از افراد دیگر مثل همکاران، دوستان یا افرادی در سایت‌های آنلاین در پیدا کردن نرم افزارهای جایگزین کمک بخواهید. این تکنیک زمانی کمک کننده است که شما از قبل یک سرچ ساده انجام داده اید اما به نتیجه دلخواه نرسیدید.

برخی از این روش‌ها، مثل استفاده از منوهای دسکتاپ، میتواند فقط نرم افزارهایی که بر روی سیستم شما نصب شده است را پیدا کند. تکنیک‌های دیگر، مثل سرچ در مرورگر، می‌توانند نرم افزارهایی که شما نصب ندارید را پیدا کند. شما معمولاً میتوانید نرم افزارها را با کمک سیستم پکیج توزیع خود نصب کنید.

## Using a Web Browser

لینوکس از مرورگر‌های متنوعی پشتیبانی می‌کند که شامل موارد زیر می‌شود:

**Chrome:** مرورگر گوگل کروم هدف سریع و آسان بودن دارد. از زمان معرفی آن در سال 2008، به سرعت محبوبیت یافت. با اینکه کروم عملاً یک پروژه تجاری است اما به صورت رایگان در دسترس است. یک نمونه اپن سورس به نام Chromium نیز در دسترس است.

**Firefox**: این برنامه در Mozilla.org در دسترس است و محبوب ترین مرورگر برای لینوکس است؛ همچنین در ویندوز و مک هم محبوبیت دارد. فایرفاکس یک مرورگر کامل است، پس مقدار زیادی از رم را اشغال میکند و ممکن است بهترین گزینه برای کامپیوترهای ضعیف و قدیمی نباشد.

**Web**: این برنامه که در اصل Wiki.genome.org/apps/web نام داشت، در دسترس است. به عنوان مرورگر دسکتاپ گنوم، برای سادگی و استفاده آسان طراحی شده است.

**Konqueror**: این برنامه KDE دو کارکرد دارد: همزمان هم مرورگر وب است و هم فایل منیجر. Konqueror در رابطه با اکثر وب پیج‌ها عملکرد خوبی دارد و نسبتاً سبک است و ارزش امتحان کردن را دارد مخصوصاً اگر از KDE Plasma استفاده میکنید. برای اطلاعات بیشتر: [.kde.org/applications/internet/org.kde.konqueror](http://kde.org/applications/internet/org.kde.konqueror)

**Lynx**: اکثر مرورگرهای وب، نرم افزارهای GUI هستند که تکست را در چندین فونت و گرافیک نمایش میدهند. لینکس (lynx.browser.org) در این مورد غیرمعمول است از آنجا که یک مرورگر متñی است. در نتیجه، اگر لینکس را در حالت متñی اجرا میکنید یا نمی‌خواهید درگیر محیط گرافیکی شوید انتخاب پرکاربردی است. لینکس همچنین به عنوان یک مرورگر متñی زمانی که وب پیج خود را میسازید کاربردی است؛ اگر یک پیج در لینکس قابل اجرا باشد، افراد نابینا قادر خواهند بود تا با تبدیل متن به گفتار از محتوای پیج استفاده کنند.

**Opera**: یک شرکت کننده غیرعادی در جمع مرورگرهای لینوکس، اپرا (Opera.com) ادعا میکند که به طور غیر معمولی سریع است. با اینکه اپرا تجاری است، اما میتوان آن را رایگان دانلود کرد.

غایب قابل توجه این لیست یک مرورگر مایکروسافت است. متاسفانه، برخی وبسایت‌ها بدون یک مرورگر مایکروسافت کار نمیکنند. باقی وبسایت‌ها ممکن است سختگیر باشند اما حداقل با یک مرورگر لینوکس کار میکنند. در نتیجه ممکن است نیاز به نصب حداقل دو مرورگر لینوکس داشته باشید.

مرورگرها میتوانند به معنای واقعی کلمه به کاربرها دسترسی آسان به دنیایی از اطلاعات بدهند. متاسفانه، وب بخش تاریکی هم دارد. مشکلات شامل زیر میشود:

- وبسایت‌ها میتوانند اطلاعات کاربر را ذخیره کنند که ممکن در بازاریابی یا راه‌هایی که مورد پسند شما نباشد استفاده شوند

- اکثر محتوای وب اتوماتیک است. یعنی وبسایت‌ها برنامه‌های کوچکی دانلود میکنند که مرورگر شما اجرا میکند. این محتوا ممکن است بی‌آزار باشد اما بیشتر و بیشتر برای فرستادن بدافزار استفاده میشود.

- وبسایت‌های فاسد ممکن است کاربر را در دادن اطلاعات حساس مثل اطلاعات مالی با تظاهر کردن به یک سایت مورد اطمینان، گول بزند. این تکنیک معروف به فیشینگ (Phishing) است.

• برخی وبسایت‌ها امن نیستند. اطلاعات رد و بدل شده ممکن است توسط یک کامپیوتر مداخله کننده شنود شوند. اکثر سایت‌ها، مخصوصاً سایت‌های بانک‌های اینترنتی و فروشگاه‌های آنلاین، اطلاعات حساس خود را رمزنگاری می‌کنند؛ اما شما نیز باید در فرستادن چنین اطلاعاتی محتاط عمل کنید.

• با خاطر دلایل امنیتی، پسورد‌های مورد استفاده در اکثر سایت‌ها سوژه سرقت است. این میتواند مشکل ساز باشد زیرا به خاطر سپردن تمام پسورد‌های وبسایت‌تان کار سختی خواهد بود. بسیاری از مرورگرها میتوانند اینکار را برای شما انجام دهند اما این باعث ذخیره پسورد روی هارد دیسک می‌شود که آسیب پذیر در برابر از دست رفتن یا دزدیده شدن است.

نکته: فصل 13. در "Creating Users and Groups" نحوه ساخت پسوردی که هم‌زمان قابل به خاطر سپردن و حدس آن سخت است را توضیح میدهد.

البته بیشتر این مشکلات در وب منحصر به فرد نیستند. برای مثال، بیشتر انتقال‌ها از طریق ایمیل نامن هستند. پس بهتر است اطلاعات حساس را از طریق ایمیل ارسال نکنید.

## Using Email Clients

نرم افزارهای ایمیل کلاینت به شما قابلیت خواندن و نوشتن ایمیل میدهند. چنین نرم افزارهایی با دسترسی به صندوق پست روی خود کامپیوتر یا با استفاده از پروتکل‌های شبکه ایمیل که بعداً توصیف خواهند شد، ایمیل را با کمک کامپیوترهای سرور شبکه ایمیل ارسال و دریافت می‌کنند. ایمیل کلاینت‌های رایج لینوکس عبارت اند از:

**Evolution**: این نرم افزار، در [wiki.gnome.org/apps/evolution](http://wiki.gnome.org/apps/evolution) یک ایمیل کلاینت GUI قدرتمند می‌باشد. همچنین شامل دفترچه آدرس و امکانات زمانبندی می‌باشد.

**KMail**: پروژه KDE در [Userbase.kde.org/kmail](http://Userbase.kde.org/kmail) به خوبی با محیط KDE Plasma سازگار شده، اما همچنان در محیط‌های دسکتاپ دیگر نیز قابل استفاده است.

**Mutt**: یکی از چندین ایمیل کلاینت‌های متنی. برخلاف اینترفیس متنی آن، Mutt بسیار کارآمد است. برای اطلاعات بیشتر میتوانید به [Mutt.org](http://Mutt.org) رجوع کنید.

**Thunderbird**: این نرم افزار در [thunderbird.net](http://thunderbird.net) یک ایمیل کلاینت وابسته به مرورگر فایرفاکس است.

ایمیل کلاینت‌ها در هر سیستم عاملی عملکرد مشابهی دارند. معمولاً، باید آنها را پیکربندی کنید تا بدانید چگونه ایمیل ارسال و دریافت کنید-یا از امکانات کامپیوتر لوکال استفاده شود یا سرور‌های ریموت. پس از آن، میتوانید ایمیل‌های دریافتی را بخوانید و ایمیل ارسال کنید.

## Using Office Tools

لینوکس چندین پکیج آفیس قابل دسترس با چند ترکیب پردازشگر کلمه، Spreadsheets، نرم افزارهای ارائه پرزنتیشن، نرم افزارهای گرافیک، دیتابیس ها و باقی برنامه ها دارد. مثال ها عبارت اند از:

**Calligra**: این مجموعه آفیس از دل یک مجموعه آفیس محبوب قدیمی KOffice به نام KOffice متولد شده است. با اینکه KOffice دیگر در دسترس نیست، کالیگرا (Calligra.org) در حال پیشرفت است. مجموعه آفیس آن شامل Word (پردازشگر کلمه)، Stage (ارائه پرزنتیشن)، Sheets (برای Spreadsheets Flow) و (فلوچارت) و Kexi (دیتابیس) میباشد. درکنار نرم افزارهای آفیس، کالیگرا نرم افزارهای گرافیکی و مدیریت پروژه نیز ارائه میدهد.

**Apache OpenOffice**: این مجموعه آفیس که در OpenOffice.org در دسترس است در گذشته OpenOffice نام داشت تا اینکه اسپانسر آن، اوراکل، در سال 2011 به Apache Group، که هم اکنون آن را نگهداری میکند اهدا کرد. امروزه نام رسمی آن Apache OpenOffice است. شش نرم افزار (پردازشگر کلمه)، Impress (ارائه پرزنتیشن)، Base (دیتابیس)، Draw (گرافیک برداری) و Calc (ادیتور معادلات) توسط این مجموعه ارائه میشوند.

**LibreOffice**: این مجموعه به عنوان شاخه ای از نسخه قدیمی پیش از آپاچی OpenOffice ساخته شده بود و در حال تبدیل شدن به محبوب ترین مجموعه آفیس لینوکس است. این مجموعه شش نرم افزار Writer، Math و Calc، Impress، Base، Draw را ارائه میدهد. مطمئناً متوجه شدید که این نرم افزارها نام یکسانی با نرم افزارهای Apache OpenOffice دارند. برای اطلاعات بیشتر میتوانید به Libreoffice.org مراجعه کنید.

نکته: شاخه یک برنامه زمانی رخ میدهد که یک پروژه به دو پروژه تقسیم شود؛ معمولاً به دلیل اینکه گروه های مختلفی از توسعه دهندگان اهداف متفاوتی داشته باشند.

اکثر این نرم افزارها از فرمت ODF (OpenDocument) پشتیبانی میکنند که یک سری از فرمت های فایل است که به آرامی به سمت تبدیل شدن به فرمت استاندارد برای پردازشگرهای کلمه، Spreadsheet، و دیگر نرم افزارهای های آفیس میروند.

بسیاری دیگر از نرم افزارها در این زمینه وجود دارند، اگرچه عضوی از یک مجموعه آفیس نیستند. برخی از آنها غیرمعمول هستند. برای مثال، Lyx میتواند جایگزین یک پردازشگر کلمه باشد، اما در یک شکل خاص برای ایجاد و ادیت فایل های LaTeX ساخته شده است. LaTeX یک فرمت فایل محبوب در علوم کامپیوتر، ریاضیات و دیگر حوزه های تکنیکال است. (Lyx.org)

## Using Multimedia Applications

لینوکس شهرت ممتازی به عنوان یک پلتفرم سرور پرکار دارد، اما ظرفیت آن به عنوان یک سیستم عامل مالتی مدیا کمبود دارد. این عموماً به دلیل غیبت اپلیکیشن های مولتی مدیا بوده است. اگرچه در چند دهه اخیر، لیست نرم افزارهای مولتی مدیا به طور قابل توجهی رشد داشته است. پیشنهاد های فعلی عبارت اند از:

**Audacity**: این ادیتور صوتی موجود در [Sourceforge.net/projects/audacity](http://Sourceforge.net/projects/audacity) به محصولات تجاری مثل Sound Forge شباهت دارد. شما میتوانید با این نرم افزار بخش هایی از یک فایل صوتی را کات کنید، ولوم را اکولایز کنید، صدای های مصنوعی اضافه کنید و خیلی امکانات دیگر.

**Blender**: با استفاده از این مجموعه خلق D3 میتوانید تصاویر سه بعدی پیچیده از جمله بی حرکت و انیمیشن بسازید. برای اطلاعات بیشتر میتوانید به [Blender.org](http://Blender.org) مراجعه کنید.

**Castero**: این کلاینت پادکست متنی به شما امکان سایسکرایب کردن در پادکست های مورد علاقهتان (و خیلی بیشتر) را میدهد، به راحتی پادکست های جدید را سرج کنید، پلی لیست ها را ببینید و کلی امکانات بیشتر. برای اطلاعات بیشتر میتوانید به [github.com/xgi/castero](http://github.com/xgi/castero) مراجعه کنید.

**GIMP**: نرم افزار ادیت عکس گنو ([Gimp.org](http://Gimp.org)) به طور کلی یک نرم افزار مشابه به Adobe Photoshop است. (کیت ابزار GTK+, که پایه و اساس گنوم و بسیاری از برنامه های دیگر بوده است، در اصل برای GIMP خلق شده بوده است).

**ImageMagick**: یک مجموعه از نرم افزارهای گرافیکی دارای یک ویژگی خاص: برنامه های ImageMagick معمولاً از طریق کامند لاین استفاده میشوند. میتوانید از آن برای تغییر فرمت فایل ها، اضافه کردن فریم به تصاویر، تغییر سایز تصاویر و خیلی اهداف دیگر استفاده میشود. برای اطلاعات بیشتر میتوانید به [Imagemagick.org](http://Imagemagick.org) مراجعه کنید.

**Open Broadcaster Software (OBS) Studio**: استریمینگ زنده از طریق OBS Studio ممکن است. همچنین، این نرم افزار امکان ضبط صدا و تصویر را بر روی دسکتاپ لینوکس فراهم میکند. امکانات قابل تنظیم زیادی توسط این برنامه قدرتمند ارائه میشود، پس برای یاد گرفتن نیاز به زمان دارد؛ همچنین با سیستم های مالتی اسکرین بهترین عملکرد را دارد. میتوانید از [obsproject.com](http://obsproject.com) آن را دریافت کنید.

نکته: برخی از فیلم هایی که با لینوکس تدوین شده اند عبارت اند از تایتانیک، مجموعه فیلم های شرک و آواتار. بسیاری از انیمیشن ها کاملاً توسط لینوکس تدوین شده اند؛ مثل Next Gen ساخته نتفلیکس، که کاملاً توسط Blender ساخته شده است.

**Kdenlive**: اگر نیاز به ادیت ویدیو هایی که با OBS ساخته اید را دارید، Kdenlive راه حل شماست ([kdenlive.org](http://kdenlive.org)). این برنامه به شما امکان انجام اعمال اولیه تا پیشرفته ادیت ویدیو را میدهد و تقریباً هرگونه فرمت صوتی تصویری میتواند در Kdenlive استفاده شود و نیازی به تبدیل فرمت ها نیست.

با وجود این طیف گسترده از ابزارهای مالتی مدیا، شما میتوانید از لینوکس برای هر کاری استفاده کنید. از کراپ کردن عکس تولد بچه دو ساله تا تدوین فیلم های بزرگ. اگر نیازهای خاصی دارید، کمی جستجو میتواند نتایج دیگری رقم بزند، این لیست فقط یک شروع است.

## Using Linux for Cloud Computing

رایانش ابری عمومی، فضای ذخیره نرم افزارهای کامپیوتر و اطلاعات اینترنت، به جای ذخیره مستقیم روی کامپیوتر است. در این عبارت، ابر به معنای اینترنت و رایانش به معنای کاری که در اینترنت انجام میدهد است. در بسیاری از موارد، کاربران به منابع رایانش ابری از طریق مرورگر وب دسترسی پیدا میکنند. در نتیجه، در تئوری، لینوکس میتواند به عنوان پلتفرم کلاینت رایانش ابری عمل کند. صرفاً یک مرورگر را برای دسترسی به سرویس دهنده رایانش ابری اجرا کنید و آغاز میشود.

در عمل، زمانی که به سرویس رایانش ابری عمومی دسترسی پیدا میکنید ممکن است پیچیدگی هایی رخ دهد. برای مثال، یک سرویس دهنده رایانش ابری ممکن است از شما بخواهد فقط از یک نوع خاص مرورگر استفاده کنید یا از پلاگین های خاصی در مرورگر خود استفاده کنید. در برخی موارد، ممکن است برآورده کردن انتظارات آنها در لینوکس غیرممکن باشد؛ با این حال، اگر سرویس دهنده طیف وسیعی از مرورگر ها را به عنوان کلاینت پشتیبانی کند، مشکلی در استفاده از منابع رایانش ابری نخواهید داشت.

برخی منابع رایانش ابری جالب توجه عبارت اند از:

- رسانه پرتقاضای استریمینگ، مثل نتفلیکس
- سرویس های ذخیره فایل، مانند Dropbox
- مجموعه های کاربردی آفیس، مثل Zoho Office
- ایمیل بر پایه وب، مانند Gmail

رایانش ابری خصوصی تکنولوژی کمی متفاوت تر است که در آن ابر، به جای اینترنت، شبکه کمپانی (یا خانه) و منابع آن است. در نتیجه، یک ابر خصوصی گاهی اوقات نسخه سازمانی یا ابر داخلی نام دارد. استفاده از این نوع ابر امنیت بیشتری فراهم میکند اما نیاز به منابع و مدیریت داخلی بیشتری می باشد.

در استفاده از لینوکس به عنوان سرور، میتوانید یک ابر خصوصی برای نگهداری فایل ها با استفاده از یکی از مجموعه نرم افزارهای زیر راه اندازی کنید:

- ownCloud ([owncloud.org](http://owncloud.org))
- Nextcloud ([nextcloud.com](http://nextcloud.com))
- FileCloud ([getfilecloud.com](http://getfilecloud.com))

این مجموعه نرم افزارها سرویسی مشابه با Dropbox ارائه میدهند، اما به جای ذخیره شدن در یک فضای ابری عمومی، معمولاً به صورت لوکال (عمل Self-Hosting) ذخیره میشوند. با این حال، می توانید در این نرم افزارهای ابری خصوصی اجزه ادغام فایل هاستینگ بر روی سیستم عامل های نصب شده بر روی سرویس دهنده های ابری دیگر مثل وب سرویس آمازون (AWS)، گوگل کلود و مایکروسافت آزور را دهید. سیستم

عامل هایی که بر روی سرویس دهنده های ابری نصب میکنید اغلب به عنوان زیرساخت به عنوان سرویس (IAAS) نام برده میشوند.

نکته: منابع رایانش ابری اغلب بر روی سرورهای متعلق به کمپانی هایی که سرویس دهنده ابری نام دارند میزبانی میشوند. یک مثال برای این موضوع اپلیکیشن محبوب توییتر است، که بر روی گوگل کلود اجرا میشود. سرویس دهنده های ابری از سرورهای قدرتمند برای ارائه ماشین مجازی های اختصاصی (سیستم کامپیوتری شبیه سازی شده که به عنوان کامپیوتر فیزیکی ظاهر و عمل میکنند) به مقاضیان خود استفاده میکنند. معمولاً، از لینوکس به عنوان سیستم عامل سرور سخت افزاری استفاده میکنند. اما چیزی که واقعا جالبه این است لینوکس اغلب سیستم عامل مورد استفاده توسط کامپیوتر مجازی مقاضیان نیز هست.

## Using Mobile Applications

با وجود اینکه اندروید یک سیستم عامل بر پایه لینوکس است، اکثر مواقع اپلیکیشن های کاملاً متفاوتی نسبت به نسخه دسکتاپ یا سرور اجرا میکند. این قابل درک است که احتمالاً شما سعی نمی کنید یک مطلب طولانی، مثل یک کتاب را با موبایل بنویسید. بسیاری از امکانات در یک نرم افزار بزرگ آفیس، مثل LibreOffice's Write، بر روی موبایل هدر می روند.

در عوض، رایانش موبایل معمولاً بر روی برنامه های کوچک به نام اپ تمرکز میکند. در مورد اندروید، میتوانید اپ ها را با استفاده از یک اپ به نام Google Play دانلود کنید. (یک نسخه وب در Play.google.com/store در دسترس است). اپ ها معمولاً محاسبات سریع و تخصصی را اغلب با به کار گرفتن امکانات گوشی موبایل فراهم میکنند. برای مثال، یک اپ میتواند مقدار کالری هایی که حین دوچرخه سواری سوزاندید را محاسبه کند یا پیش بینی آب و هوای منطقه شما را دریافت کند. هر دوی این مثال ها از امکانات GPS گوشی شما استفاده میکنند تا موقعیت گوشی (و شما) را شناسایی کنند.

با اینکه اکثر اپلیکیشن های لینوکس برای دسکتاپ و سرور اپن سورس و به صورت رایگان در دسترس هستند، برخی اپ های اندروید رایگان نیستند. پس قبل از دانلود یک اپ از هزینه آن اطمینان حاصل کنید.

نکته: اپ های اندروید به طور فزاینده منشا بدافزارها هستند. میتوانید با دانلود از گوگل پلی یا دیگر اپ استورهای مورد اعتماد ریسک خود را به حداقل برسانید.

## Using Server Programs

لینوکس یک سیستم عامل قدرتمند برای اجرای برنامه های سرور است؛ پس تعجب برانگیز نخواهد بود وقتی طیف گسترده ای از برنامه های سرور برای لینوکس پیدا می کنید. در این بخش، برخی از پروتکل های رایج سرور و برنامه هایی که از آنها استفاده میکنند توضیح داده خواهند شد؛ همچنین پروسه نصب و راه اندازی سرورها همراه با اطلاعات پایه در رابطه با مشکلات امنیتی سرور نیز پوشش داده می شود.

## Identifying Common Server Protocols and Programs

شبکه ها، از جمله اینترنت، به وسیله پروتکل ها عمل میکنند. پروتکل های شبکه توصیفات شفافی هستند که دو کامپیوتر چگونه باید با هم تبادل اطلاعات کنند تا به نتیجه خاصی برسند؛ مانند انتقال ایمیل یا تحويل یک فایل برای پرینت شدن. اکثر پروتکل ها در یک داکیومنت استانداردها یا بیشتر توصیف شده اند که با نام RFC Request for Comments یا RFC شناخته میشوند، که هرکدام یک شماره دارند. معمولاً یک داکیومنت RFC پروتکل را تعریف میکند، و در طول زمان داکیومنت های RFC الحاقی، ملحقات و اصلاحات را از آنجایی که واجب می شوند توضیح میدهد.

اکثر پروتکل های شبکه شامل انتقال اطلاعات به یک و بیش از یک پورت میشوند که منابع شماره گذاری شده مشخصی بر روی یک کامپیوتر هستند. میتوانید پورت ها را به عنوان شماره اتاق در یک ساختمان در محوطه دانشگاه در نظر بگیرید. شماره اصلی (یک اینترنت پروتکل یا آدرس IP) کامپیوتر را به طور کلی تعیین هویت میکند، و شماره پورت پروتکل مورد استفاده را تعیین هویت میکند. یک برنامه سرور خود را به یک شماره پورت متصل میکند و تمام درخواست های ورودی را از آن پورت دریافت میکند.

جدول 4.1 برخی شماره پورت های رایج، پروتکل های مربوط به آنها و برنامه لینوکسی که اغلب در ارتباط با این پروتکل ها استفاده میشود را خلاصه بندی میکند. بیشتر پورت ها و پروتکل ها با بیش از یک برنامه کار میکنند. این به این دلیل است که لینوکس گزینه های زیادی برای پروتکل های زیادی فراهم میکند؛ شما میتوانید کدام یک از چندین برنامه سرور را برای پروتکل مورد نظر استفاده کنید، همانطور که میتوانید انتخاب کنید کدام یک از چندین پردازشگرهای کلمات یا مرورگرهای وب را استفاده کنید.

نکته: فایل /etc/services پورت های رایج را به نام های کوتاهی که اغلب در فایل های پیکربندی استفاده می شوند ربط میدهد.

Table 4.1 Common port numbers and their purposes

Port number	Protocol	Common Server program(s)	Explanation
20-21	FTP	ftpd, ProFTPD, Pure-FTPD, vsftpd	پروتکل انتقال پروندها (FTP) یک پروتکل قدیمی برای انتقال فایل ها در شبکه است. این پروتکل هم از دسترسی ناشناس و هم از دسترسی توسط رمز عبور پشتیبانی میکند. FTP با استفاده از دو پورت، در مقابل سایر پروتکل ها غیر معمول است.

22	SSH	OpenSSH	شل امن (SSH) یک ابزار دسترسی از راه دور رمزنگاری شده است. همچنین از انتقال فایل و رمزنگاری سایر پروتکل‌ها پشتیبانی می‌کند.
23	Telnet	telnetd	تلنت یک پروتکل قدیمی و بدون رمزنگاری برای لاگین از راه دور به سیستم است. امروزه به ندرت استفاده می‌شود، اگرچه برنامه کلاینت آن، تلننت، می‌تواند به عنوان یک ابزار تشخیص خطأ در شبکه مفید باشد.
25	SMTP	Postfix, qmail, sendmail	پروتکل انتقال ساده ایمیل (SMTP)، پروتکل اصلی برای انتقال ایمیل در اینترنت است. فرستنده انتقال‌های SMTP را آغاز می‌کند.
53	DNS	dnsmasq, named	سامانه نام دامنه (DNS) به کامپیوترها این امکان را می‌دهد که با فراهم کردن نام میزبان (hostname)، یک آدرس IP را جستجو کنند یا برعکس. بدون آن، شما باید تمامی کامپیوترها را به جای استفاده از نام با استفاده از آدرس IP ارجاع دهید.
67	BOOTP, DHCP	dnsmasq, dhcpd	پروتکل بوتاسترپ (BOOTP) و فرزنده کوچکتر آن، پروتکل پیکربندی میزبان دینامیک (DHCP)، هر دو امکان پیکربندی خودکار یک کامپیوتر در شبکه محلی را فراهم می‌کنند تا به صورت خودکار دیگر کامپیوترها را برای استفاده از شبکه پیکربندی کنند.
80	HTTP	Apache HTTPD, NGINX	پروتکل انتقال هایپر تکست اساس وب جهانی (HTTP) یا به طور ساده وب (WWW)

			است.
109-110	POP2 and POP3	Courier, Cyrus IMAP, Dovecot, UW IMAP	پروتکل دفتر پستی (POP) از چندین نسخه گذر کرده است، هر کدام با پورت خود. این پروتکل به یک گیرنده امکان می‌دهد تا انتقال ایمیل را آغاز کند، بنابراین اغلب به عنوان آخرین مرحله در تحويل ایمیل، از سرور به گیرنده، استفاده می‌شود.
118	SQL	MySQL, PostgreSQL, MariaDB	زبان پرس‌وجوی ساختاری (SQL) یک زبان رابط دیتابیس فعل در شبکه است. اگر یک سرور SQL را در شبکه خود اجرا کنید، کامپیوترهای کلاینت می‌توانند به این دیتابیس دسترسی پیدا کنند و آن را تغییر دهند.
137-139	SMB/CIFS	Samba	و SMB مايكروسافت از پروتکل CIFS برای اشتراك فايل و پريнтер استفاده مي‌کند و سامبا اين پروتکل ها را در لينوكس جاي گذاري مي‌کند
143, 200	IMAP	Courier, Cyrus IMAP, Dovecot, UW IMAP	پروتکل دسترسی پیام اینترنتی (IMAP) یک پروتکل انتقال ایمیل است که توسط گیرنده آغاز می‌شود، مشابه پروتکل POP. با این حال، IMAP به گیرنده‌گان این امکان را می‌دهد که ایمیل‌ها را به طور دائمی در کامپیوتر سرور ذخیره و مدیریت کنند.
389	LDAP	OpenLDAP	پروتکل دسترسی سبک به دایرکتوری (LDAP) یک پروتکل شبکه برای دسترسی به دایرکتوری‌ها است که در این زمینه نوعی از دیتابیس هستند. اغلب برای ذخیره LDAP

			اطلاعات لاینین به سیستم شبکه و سایر موارد استفاده می‌شود.
443	HTTPS	Apache HTTPD, NGINX	این پروتکل یک نسخه امن (رمزنگاری شده) از HTTP است.
2049	NFS	NFS	سامانه فایل شبکه (NFS) یک پروتکل و یک سرور با همین نام برای به اشتراک گذاری فایل بین سیستم‌عامل‌های یونیکس و شباهت‌های یونیکس

جدول 4.1 ناقص است؛ این جدول صرفاً برخی از پروتکل‌های مهم تر و سرورهایی که آنها را ارائه می‌دهد را خلاصه بندی می‌کند. پروتکل‌ها و سرورهای بیشمار دیگری وجود دارند و بسیاری از آنها امور بسیار تخصصی ای انجام می‌دهند.

نکته: در فصل 15، پیکربندی شبکه در جزئیات گسترده‌تری شرح داده می‌شود.

برخی پروتکل‌ها بیشترین استفاده را در شبکه‌های لوکال دارند. برای مثال، DHCP ذاتاً با هدف کمک در مدیریت شبکه لوکال با آسان تر کردن پیکربندی کامپیوترهای کلاینت بوجود آمده است- صرفاً به کامپیوترها بگویید که از DHCP استفاده کنند و همین و بس. SMB/CIFS نیز معمولاً در شبکه لوکال به کار گرفته می‌شود تا ایجاد امکان دسترسی به فایل‌ها و پرینتر یکدیگر را آسان تر کند. و اما پروتکل‌هایی مثل HTTPS، عموماً در اینترنت به طور کلی استفاده می‌شوند، با اینکه میتوانند در شبکه لوکال نیز استفاده شوند.



### Real World Scenario

## Server Programs and Server Computers

اصطلاح سرور میتواند برای یک کامپیوتر کامل یا یک برنامه واحد درحال اجرا روی آن کامپیوتر به کار رود. زمانی که برای کامپیوتر به کار می‌رود، هدف کامپیوتر و این موضوع که بیش از یک برنامه سرور بر روی آن اجرا می‌شود را مشخص می‌کند. کامپیوترهای سرور معمولاً سرویس‌هایی فراهم می‌کنند که در هر جایی توسط میلیون‌ها کامپیوتر کلاینت استفاده می‌شوند. کامپیوتر کلاینت به کامپیوترهایی گفته می‌شود که از سرویس‌های یک سرور استفاده می‌کنند.

در دنیای شبکه، یک سرور (کامپیوتر یا برنامه) منتظر اتصال از طرف کلاینت (کامپیوتر یا برنامه) است و به درخواست‌های انتقال اطلاعات پاسخ می‌دهد. کامپیوترهای سرور اغلب، اما نه همیشه، از کلاینت‌های خود قدرتمند تر هستند.

زمانی که کلمه سرور را میخوانید (یا کلاینت، از آن باب)، ممکن است منظورش یک کامپیوتر یا یک برنامه باشد. خود جمله معمولاً مشخص میکند که منظور کدام معنی است؛ با اینکه گاهی اوقات موضوع این نیست، در واقع، گاهی گوینده یا شنونده ممکن است نداند! برای مثال، شخصی ممکن است گزارش دهد "سرور سامبا از کار افتاده." در این مورد، شما باید بفهمید آیا مشکل از برنامه سرور سامبا است یا چیز دیگری در کامپیوتر سرور باعث اشکال شده است.

گاهی اوقات، ممکن است مرز بین سرور و کلاینت تار شود. برای مثال، در تنظیمات آفیس، برای برخی کامپیوتراها رایج است که با اجرا کردن نرم افزار فایل سرور مثل Samba یا NFS به عنوان فایل سرور عمل کنند. چنین پیکربندی هایی این امکان را ایجاد میکند تا دو نفر فایل های خود را برای یکدیگر در دسترس قرار دهند. در چنین وضعیتی، هر دو کامپیوتراها هم‌زمان به عنوان کلاینت و سرور عمل میکنند و هر دو یک نوع نرم افزار را اجرا میکنند. اما در هرگونه معاوضه ای، تنها یکی کلاینت و دیگری سرور است.

## Focusing on Web Servers

یک سرور وب صفحات وب را تحويل کاربران داخلی یا خارجی شبکه میدهد. اگر تا به حال از وب جهان استفاده کرده باشید، به احتمال زیاد از دو وب سرور محبوب که در لینوکس عرضه میشوند استفاده کرده اید:

**Apache HTTPD**: سرور آپاچی HTTPD بخش از لینوکس مرسوم، آپاچی، PHP (LAMP) Stack و MySQL برای اپلیکیشن های وب است. پکیج نرم افزاری اصلی وب سرور در سال 1995 منتشر شد. در کمتر از یک سال، آپاچی تبدیل به محبوب ترین وب سرور اینترنت شد. تا به امروزه آپاچی HTTPD نه تنها برای لینوکس، بلکه برای لطف پایداری و قابل اعتماد بودن خود حفظ کند. سرور آپاچی HTTPD نه تنها برای لینوکس، بلکه برای یونیکس، BSD، ویندوز و حتی مکینتاش در دسترس است. برای اطلاعات بیشتر میتوانید به <http://httpd.apache.org> مراجعه کنید.

**NGINX**: منتشر شده در سال 2002، وب سرور NGINX (تلفظ X Engine)، سروری نسبتاً تازه وارد به بازار است. NGINX میتواند از طرف کلاینت منابع یک یا بیش از یک سرور را دریافت کند و هم‌زمان به عنوان وب سرور عمل کند. به خاطر همین امکانات، و این موضوع که سریع و سبک است، NGINX برخی وبسایت های عظیم مثل نتفلیکس را برای خود بدست آورده است. میتوانید با مراجعه به [nginx.org](http://nginx.org) اطلاعات بیشتری کسب کنید.

بهترین قابلیت این دو وب سرور این است که مجبور نیستید حتماً یکی را انتخاب کنید. بسیاری از ادمین های سرور نصب دوگانه را انتخاب میکنند، و هر دوی NGINX و Apache HTTPD را استفاده میکنند. گاهی اوقات یک معماری پهلو به پهلو اجرا میشود، به این صورت که هر سرور کاری که در آن بهترین است را انجام میدهد—آپاچی محتوای دینامیک و NGINX محتوای استاتیک را مدیریت میکنند (برای هر کاربر به یک شکل). بعضی دیگر معماری «آپاچی در عقب» (یا NGINX در جلو) را به کار میگیرند که این امکان را ایجاد میکند که NGINX سرویس های بازیابی منابع خود را به کار گیرد و آپاچی پایدار همچنان محتوای دینامیک را بر حسب نیاز فراهم کند.

## Installing and Launching Servers

بحث نگهداری برنامه های سرور فراتر از حیطه این کتاب است، اما بهتر است از اصول اولیه این امر آگاه باشید. میتوانید سرورها را به همان روش نصب نرم افزارهای دیگر نصب کنید، با روشی که در این فصل و با جزئیات بیشتر در فصل ۹ توصیف می شود.

پس از آنکه نرم افزار نصب شد، باید سرور را اجرا کنید. این کار با اجرای اپلیکیشن های دسکتاپ متفاوت است. به جای کلیک بر روی آیکون یا ورودی منو در GUI، معمولاً باید کامپیوتر خود را به طوری پیکربندی کنید که به صورت خودکار هر زمانی که کامپیوتر بوت می شود اجرا شود. پس از آن، برنامه سرور در پس زمینه اجرا میشود به عنوان یک Daemon، که به پروسه ای میگویند که به صورت خودکار اجرا میشود.

نکته: کلمه *Daemon* از اساطیر یونان مشتق میشود؛ *Daemon* ها موجودات مفید سوپرنچرال بودند، درست مثل *Daemon* های یونیکس و لینوکس که برنامه های مفیدی هستند.

اکثر سرورها زمانی که لینوکس بوت میشود به شکل خودکار اجرا میشوند. همچنین میتوانید با باز کردن ترمینال و تایپ کردن یک دستور متنی مثل `start` یا `stop` به صورت دستی سرور را روشن و خاموش کنید. ذات روشن شدن سرور در توزیع های اخیر در حال تغییر است، و بحث آن فراتر از گستره این کتاب است. با این حال، خوب است بدانید که توزیع های گوناگونی از روش آماده سازی Daemon خاصی برای استارت و مدیریت Daemon های سرور های متتنوع استفاده می کنند. میتوانید با مطالعه اسناد راهنمای توزیع خود متوجه شوید که توزیع شما از کدام روش آماده سازی Daemon در لیست زیر استفاده میکند:

- System V init
- systemd
- Upstart

برخی سرورها توسط یک ابر کامپیوتر اجرا میشوند؛ مثل `xinetd`. این برنامه های سرور دائماً در حال اجرا هستند و سرورهای تحت مدیریت خود را خالی نگه می دارند مگر زمانی که نیاز باشد. این پیکربندی میتواند تاثیر اجرای دائمی مموری را در سرورهایی که به ندرت استفاده می شوند به حداقل برساند. ابر سرور همچنین میتواند با اهداف امنیتی عمل کند؛ مثل `Bouncer`، و خرابکاران را دور نگه دارد.

## Securing Servers

هر زمانی که شما یک سرور را اجرا میکنید، همzمان ریسک به خطر افتادن و سواستفاده آن از را هم ایجاد میکنید. ریسک ها شامل دسته بندی های زیر میشوند:

• سرورها میتوانند حاوی باگ هایی باشند که به دیگران اجازه سو استفاده از نرم افزار برای اجرای لوکال برنامه ها را دهد.

• ممکن است شما سرور را به خوبی پیکربندی نکنید، و به بیگانگان اجازه دسترسی بیش از حد مورد نظرتان به سیستم خود دهید.

• کاربرانی که دارای اکانت با دسترسی ریموت از طریق سرور را دارند میتوانند از اعتمادی که به آنها شده سواستفاده کنند. این ریسک میتواند به طرز قابل توجهی شدیداً بزرگتر باشد زمانی که با بگ سرور یا پیکربندی غلط ترکیب شود.

• یک سرور میتواند به عنوان یک سکو برای حمله به دیگران استفاده شود که باعث میشود شرایط جوری به نظر بیاید که انگار منشا حمله از کامپیوتر شما بوده است.

• حتی بدون نفوذ به یک کامپیوتر، یک اتکر می‌تواند سرور را در اطلاعات جعلی غرق کند که در نتیجه از کار می‌افتد. این تکنیک Denial of Service Attack) DoS نام دارد.

امنیت سرور یک بحث شدیداً پیچیده ای می‌باشد و جزئیات آن در هر سرور متغیر است. برای مثال، اگر یک سرور مثل سرور لاجین ریموت، سامبا، ایمیل سرور POP یا IMAP را اجرا کنید، به احتمال زیاد باید به امنیت پسورد با دقت توجه کنید، از آنجایی که تمام سرورها بر پسوردها تکیه میکنند. البته پسورددها برای سرورهای DNS و DHCP فاقد اهمیت هستند. با این حال، حتی اگر یک برنامه سرور DHCP یا DNS از پسورد استفاده نکنند، باقی برنامه های سرور که در کامپیوتر در حال اجرا هستند از پسورد استفاده میکنند.

به طور کلی، ایمن کردن یک سرور نیاز به توجه به هر یک از فاکتورهای ریسک ذکر شده دارد. گام های مشخصی که میتوانید برای ایمن کردن سرورهای خود بردارید عبارت اند از:

• بهتر است همیشه برنامه سرور خود را با استفاده از ابزارهای مدیریت پکیج برای آپگرید سرور زمانی که آپدیتی در دسترس است به روز نگه دارید. همچنین میتوانید در رابطه با سرورها تحقیق کنید و آنهای که بهترین امنیت را دارند انتخاب کنید.

• در رابطه با پیکربندی سرور مطالعه کافی داشته باشید تا مطمئن شوید سرورهای خود را به درستی پیکربندی میکنید.

• بهتر است اکانت های بی استفاده را حذف کنید و اکانت های لازم را بررسی کنید که حتماً پسورد های قدرتمندی داشته باشند.

• میتوانید از پیکربندی های فایروال برای محدود کردن دسترسی دیگران به کامپیوتراهای سرور که فقط هدف استفاده داخلی دارند استفاده کنید. همچنین میتوانید از فایروال برای به حداقل رساندن ریسک تبدیل شدن کامپیوتراهای خود به وسیله حمله به دیگران استفاده کنید.

## Managing Programming Languages

بسیاری از کاربران هرگز مجبور به سر و کله زدن با زبان های برنامه نویسی نیستند؛ با این حال، اطلاعات پایه از چیزی که هستند و چه فرقی با یکدیگر دارند برای کاربران لینوکس به دلایل مختلف اهمیت دارد. ممکن است نیاز به نصب زبان های برنامه نویسی برای کاربران روی سیستم هایی که مدیریت می کنید یا برای خودتان برای کامپایل کردن نرم افزار از سورس کد داشته باشد. همچنین شاید قصد یادگیری برنامه نویسی را داشته باشید، مخصوصاً اگر قصد اتوماتیک کردن امور مدیریتی کامپیوتر از طریق اسکریپت شل دارید.

این بخش اطلاعات اولیه در رابطه با زبان های برنامه نویسی را ارائه میدهد. با توضیح تفاوت های بین کامپایل شده و زبان تفسیری آغاز میشود که درک آنها بسیار مهم است تا بتوانید به طرز درست فایل های برنامه را مدیریت کنید یا انتخاب کنید از چه چیزی میخواهید استفاده کنید. همچنین توضیح مختصری از برخی زبان های رایج برنامه نویسی فراهم شده است تا بتوانید آنها را تشخیص دهید و از فایل های سورس کد آنها استفاده کنید یا تصمیم بگیرید کدام زبان را میخواهید برای استفاده فرا بگیرید.

## Choosing Compiled Vs an Interpreted Language

در هسته مرکزی خود، کامپیوتر ها کد های باینری را میفهمند—اعدادی که نمایانگر عملیات ها مثل جمع دو عدد یا انتخاب بین دو عمل برای انجام هستند. با این حال مردم، در کار با کلمات و سمبل ها مثل + یا if بسیار تواناتر هستند. در نتیجه اکثر زبان های برنامه نویسی شامل نوشتن یک برنامه در یک زبان سمبولیک و بعد ترجمه آن سمبل ها به شکل عددی برای فهم کامپیوتر می باشند. ده ها، اگر نه صد ها، از چنین زبان های برنامه نویسی ای وجود دارد، که هرکدام ویژگی های خاص خود را دارند.

در میان زبان های سطح بالا، دو دسته بنده کلی وجود دارد:

**زبان های کامپایل شده:** برنامه نویس ها برنامه ای که با زبان سطح بالا نوشته شده را از سورس کد اورجینال خود به شکل کد ماشین تبدیل (یا کامپایل) میکنند. پرسه کامپایل کردن ممکن است کمی زمان ببرد، معمولاً چند ثانیه تا چندین ساعت، بسته به حجم برنامه و سرعت کامپیوتر. کامپایل کردن همچنین می تواند توسط اروهایی در برنامه با شکست مواجه شود. زمانی که کامپایل کردن با موفقیت به پایان برسد، کد ماشین که در نتیجه حاصل میشود به سرعت اجرا میشود.

**زبان های تفسیری:** برنامه هایی که با زبان های تفسیری نوشته میشوند زمانی به کد ماشین تبدیل می شوند مه توسط برنامه ای به نام *Interpreter* یا مفسر اجرا شوند. عمل تبدیل بر مبنای خط به خط انجام میشود. به این معنا که برنامه کاملاً تبدیل به کد ماشین نمیشود؛ مفسر تشخیص میدهد هر خط چه کاری انجام میدهد و سپس آن کار را انجام میدهد. این یعنی برنامه های تفسیر شده بسیار کندتر از برنامه های کامپایل شده اجرا میشوند. فایده آن در این است که زبان های تفسیری راحت تر توسعه داده میشوند، از آنجایی که نیاز به پرسه کامپایل کردن نخواهند داشت. همچنین برنامه های تفسیر شده آسان تر اصلاح میشوند؛ صرفاً برنامه در یک ادیتور اپن سورس باز کنید و سپس دوباره ذخیرش کنید. این ویژگی زبان های تفسیری را بسیار مفید

برای تغییر عملیات های استارت آپ سیستم که ادمین سیستم می خواهد تغییر دهد می کند--ادمینستورها میتوانند به سرعت تغییرات را ایجاد و تست کنند.

### برنامه نویسی با اسembly:

به علاوه زبان های کامپایل شده و تفسیری، یک گزینه دیگر زبان اسembly است. این یک زبان با مطابقت یک به یک ساده بین اعداد کد ماشین و سمبول های مورد استفاده برنامه نویس است. زبان اسembly بسیار سطح پایین است، که به این معناست که یک برنامه نویس ماهر اسembly میتواند برنامه های فشرده و کارآمد تولید کند. اگرچه زبان اسembly زیاد پرتابل نیست؛ رحمت زیادی لازم است تا یک برنامه ای که برای x 64-86CPU نوشته شده را برای اجرا بر روی پردازنده ARM تبدیل کرد. همچنین نوشتن برنامه های اسembly دشوار تر از نوشتن برنامه با زبان های سطح بالا است. به همین دلیل، برنامه های زبان اسembly با قدرتمند تر شدن کامپیوترها به تدریج نادر تر شده اند؛ مزیت سرعت و حجم زبان اسembly دیگر در قرن بیست و یکم برای اکثر اهداف گیرا نیستند.

در تئوری، اکثر زبان ها میتوانند در یکی از فرم های کامپایل شده یا تفسیری استفاده شوند. اما در عمل، اکثر زبان ها معمولا فقط میتوانند در یکی از فرم ها به کار گرفته شوند.

برخی زبان ها به سادگی در هیچ کدام از دسته بندی ها قرار نمی گیرند. بخش "برنامه نویسی با اسembly" را برای یک استثنای مهم مطالعه کنید. برخی دیگر در مرز دو دسته قرار میگیرند، مثل جاوا، که از سورس کد به فرم یک پلتفرم مستقل کامپایل شده است که نیازمند تفسیر است.

## Identifying Common Programming Languages

لینوکس از طیف گسترده ای از زبان های برنامه نویسی پشتیبانی میکند، که عبارت اند از:

اسembly: همانطور که پیشتر ذکر شد، این زبان سطح پایین میتواند برنامه های کارآمدی تولید کند، اما زبان دشواری است و پرتابل نیست. در واقع، صحبت از اسembly به نحوی که انگار تنها یک زبان است کمی گمراه کننده است، از آنجایی که هر معماری CPU زبان اسembly خاص خودش را دارد

زبان C: این زبان احتمالا مهم ترین زبان کامپایل شده برای لینوکس است، از آنجایی که اکثر کرنل لینوکس، همچون تعداد زیادی از اپلیکیشن های لینوکس با زبان C نوشته شدند. C میتواند کدهای نسبتا کارآمدی تولید کند، اما همچنین نوشتن کدهای باگ دار در C به دلیل کمبود امکانات چک کردن ارورها که در خیلی زبان ای رایج است، به مراتب رخ میدهد. معمولا نام فایل های سورس کد C با .c یا .h .c .h فایل های اصلی سورس کد هستند، در حالی که فایل های .c فایل های .h هدر نام دارند، که حاوی تعریف توابع در فایل های .c هستند، برای رفرنس توسط فایل های دیگر در یک برنامه. یک برنامه بزرگ میتواند شامل ده ها، اگر نه صدها یا هزاران، فایل سورس کد بخصوص باشد. در لینوکس، برنامه های C عموما با برنامه gcc کامپایل می شوند، که بخشی از پکیج مجموعه کامپایلر گنو می باشد.

نکته: با اینکه کرنل لینوکس اکثرا با C نوشته شده است، بخش هایی از آن با زبان اسمبلی نوشته شده اند.

**زبان C++:** یک توسعه از زبان C است که امکانات شئ گرایی را اضافه میکند، یعنی تاکید بیشتری بر ساختار داده و تعاملات آنها نسبت به روش مورد استفاده برای کنترل جریان برنامه داده می شود. بسیاری از برنامه های پیچیده لینوکس، مثل LibreOffice Apache OpenOffice و KDE در لینوکس، عمدها با زبان C++ نوشته شده اند. نام فایل های سورس کد C++ معمولاً با .cxx یا .cpp یا .c یا .h یا .hxx یا .hpp یا .h یا .hh می شوند و فایل های هدر با .h یا .hxx یا .hpp یا .h یا .hh می شوند. در لینوکس، C++ عموماً با برنامه g++ کامپایل می شود، که بخشی از GCC است.

**جاوا:** جاوا توسط کمپانی Sun Microsystems (امروزه تحت مالکیت Oracle) به عنوان یک زبان بین پلتفرمی که جایی بین کامپایل شده و تفسیری می باشد خلق شد. جاوا به عنوان زبانی برای اپلیکیشن های کوچک که توسط وبسایت ها عرضه می شوند محبوب شد؛ با اینکه برخی برنامه های دیگر نیز بر پایه جاوا هستند. فایل های سورس کد جاوا معمولاً اسم .java را در انتهای خود دارند.

**جاوا اسکریپت:** جاوا و جاوا اسکریپت معمولاً با یکدیگر اشتباه گرفته می شوند، اما از خیلی جهات متفاوت هستند. یک تفاوت این است که جاوا اسکریپت یک زبان تفسیری است. همچنین، یکی از محبوب ترین زبان های برنامه نویسی وب سایت است، بسیار رایج تر از جاوا. جاوا اسکریپت در کنار HTML و CSS کار میکند تا اکثریت وب پیج های تعاملی، همگی از برنامه های جاوا اسکریپت مشتق شده اند. تقریباً تمام مرورگرهای باکس، یا نقشه های تعاملی، همگی از جاوا اسکریپت می باشند. نام فایل سورس کد جاوا اسکریپت با .js تمام می شود.

**زبان Perl:** یک زبان تفسیری طراحی شده برای ایجاد تغییرات در متن، اما چندمنظوره که میتواند برای بسیاری از کارها استفاده شوند. برنامه های Perl معمولاً در آخر نام فایل های خود .pm یا .pl یا .t دارند.

**زبان PHP:** پیش پردازنده مافوق متن، یا PHP (یک مخفف بازگشتی)، برای استفاده در وب سرورها به منظور تولید محتوای دینامیک، که محتوای متغیر بسته به کاربر، زمان روز، یا معیار دیگری میباشد ساخته شد. PHP یک زبان تفسیری است، و نیازمند وب سرور آگاه به PHP مثل آپاچی میباشد. با چنین سروری و پیکربندی مناسب، یک وبسایت میتواند از لاگین کاربران، سبدهای خرید، محتوای متفاوت بسته به موقعیت مکانی کاربر و غیره را پشتیبانی کند. نام فایل های PHP اکثراً .php می شوند با اینکه متغیرهای دیگری نیز رایج می باشند.

**پایتون:** زبان تفسیری پایتون، خوانا بودن کد را هدف اصلی خود میکند. پایتون از شئ گرایی پشتیبانی (اما ملزم نمی داند) میکند. معمولاً برای اعمال اسکریپت نویسی استفاده میشود، اما میتواند برای نوشتن برنامه های پیچیده تر نیز استفاده شود. برنامه های پایتون معمولاً از .py در آخر نام خود استفاده میکنند، با اینکه چندین متغیر دیگر نیز رایج است.

نکته: نام زبان برنامه نویسی پایتون رفرنسی به سریال کالت بریتانیایی به نام *Monty Python's Flying Circus* می باشد.

ایجاد میکنند، زبان های تفسیری خودشان را فراهم میکنند. از اینها، شل Bourne again است، پس اسکریپت نویسی Bash بسیار رایج است. خیلی از فایل ها که پروesse استارت آپ لینوکس را کنترل میکنند در واقع اسکریپت های Bash هستند. چنین اسکریپت هایی اغلب پسوند خاصی در آخر نام خود ندارند، اگرچه برخی از پسوند sh. استفاده میکنند.

نکته: فصل 11، اصول اولیه ساخت یا اصلاح اسکریپت های Bash را پوشش میدهد.

## Handling Software Packages

نصب برنامه ها روی یک توزیع لینوکس در طی سالها آسان تر شده است. با این حال، نحوه پکیج شدن نرم افزار، نصب شدن و مدیریت آن میتواند در هر توزیع بسیار متفاوت باشد. مهم است که این تفاوت ها را برای استفاده بهینه از برنامه های ذکر شده لینوکس در این بخش بدانید. این بخش صرفا توضیحات مختصراً را بهینه از نصب و مدیریت پکیج های نرم افزار در فصل 9 فراهم شدند.

## Understanding Software Package

در لینوکس، برنامه های نرم افزار در یک پکیج قرار می گیرند که نصب و مدیریت آنها را راحت تر کرده است. پکیج ها در لینوکس توسط سیستم مدیریت پکیج (PMS)، که در فصل 9 راجع به آن صحبت خواهد شد مدیریت میشوند.

این پکیج ها در ریپازیتوری ها ذخیره میشوند، که سرورهای ذخیره سازی رسمی در اینترنت هستند. ریپازیتوری ها در اینترنت از طریق ابزارهای PMS لوکال سیستم لینوکس شما قابل دسترسی هستند. توسعه دهنده های نرم افزاری زیادی در خود ذخیره دارند، که آماده کشف یا نصب هستند. در نتیجه، در اکثر مواقع، بهتر است تا برنامه ها را از ریپازیتوری دیفالت توزیع خود دریافت کنید. خوشبختانه، PMS توزیع شما به صورت دیفالت خودش این کار را انجام میدهد.

## Identifying Common Package Tools

هر توزیع از PMS و ابزارهای پکیج خاص خود استفاده میکند، که در فصل 9 با جزئیات بیشتر به آن پرداخته میشود. موارد زیر تعدادی از ابزارهای اصلی مورد استفاده توسط PMS های بزرگ هستند:

dpkg: یک ابزار پکیج سطح پایین که به عنوان فونداسیون ابزارهای PMS بر پایه دنبیان استفاده شد. این ابزار می تواند نصب مستقیم، مدیریت، و حذف پکیج های نرم افزار را انجام دهد. با این حال، در عملکرد خود محدود است. برای مثال، ابزار dpkg نمیتواند از ریپازیتوری ها پکیج نرم افزار دانلود کند.

: این ابزار نیز یک ابزار پکیج سطح پایین است که عملکرد مشابهی با `dpkg` دارد. با این حال، به عنوان `rpm` فونداسیون سیستم مدیریت پکیج لینوکس Red Hat استفاده شد. با اینکه میتوانید از `rpm` برای مدیریت پکیج ها استفاده کنید، بهتر است که از یک ابزار PMS سطح بالا استفاده کنید.

: این یک ابزار متنی برای PMS دبیان است. با `apt-get`، میتوانید از ریپازیتوری ها دانلود و نصب، و حذف پکیج های نرم افزار از سیستم لینوکس لوکال خود انجام دهید. علاوه بر این، میتوانید هر پکیج خاص، تمام پکیج های روی سیستم، یا کل توزیع را آپگرید کنید. با این حال، باید از ابزار متنی `apt-cache` برای مشخص کردن تکه های مختلف اطلاعات در رابطه با پکیج های نرم افزار استفاده کنید.

: این یک ابزار متنی برای PMS های Red Hat می باشد. از این ابزار در توزیع هایی مثل RHEL، Fedora و CentOS استفاده میشود. با `yum`، میتوانید از ریپازیتوری ها دانلود و نصب، حذف پکیج های نرم افزار از سیستم خود، آپگرید پکیج ها و خیلی بیشتر انجام دهید. علاوه بر این، میتوانید از `yum` برای مشخص کردن تکه های مختلف اطلاعات پکیج ها و مدیریت آنها، مثل نمایش یک لیست از ریپازیتوری های پیکربندی شده PMS استفاده کنید.

# Chapter 5

## Getting to Know the Command Line

ممکن از به کامند لاین به چشم یک عتیقه از دهه 70 میلادی نگاه کنید که ربطی به رایانش امروزی ندارد. اما چنین نیست! با اینکه لینوکس برنامه های **ال-U** بیشماری دارد، اکثر آنها صرفا فرانت اند های ظاهری از ابزار متنی هستند. با فراگیری ابزارهای دستور متنی، می توانید قدرت واقعی لینوکس را آزاد کنید، که به شما امکان انجام سریعتر کارهایتان را میدهد. شما همچنین قادر خواهید بود درصورت از کار افتادن سیستم **ال-U** لینوکس همچنان مدیریت کنید، یا بتوانید به صورت ریموت لاگین، و سیستم را مدیریت کنید. ابزارهای کامند لاین قادر به اسکریپت شدن نیز هستند؛ یعنی میتوانید یک برنامه ساده بنویسید که یک عملیات را سریعتر یا راحت تر از برنامه های استاندارد انجام میدهد. به همین دلایل، اکثر فصل های این کتاب روش های **ال-U** و دستور متنی انجام کارها را توضیح میدهد.

برای شروع عملیات های کامند لاین، باید بدانید چگونه شروع کنید. برای این کار، باید بدانید چگونه برنامه ها را اجرا کنید و چگونه راهنمایی بگیرید. همچنین باید با تعدادی از امکانات بهینه ساز کامند لاین های لینوکس آشنا شوید.

### Starting a command line

یک کامند لاین لینوکس، یا در حقیقت **Shell** یک برنامه است. درست مثل هر برنامه دیگری، شل باید اجرا شود. شل را میتوانید در یک پنجره **ال-U** به نام برنامه **Terminal** اجرا کنید، یا میتوانید به صورت لوکال از طریق کنسول متنی به کامپیوتر خود لاگین کنید. علاوه بر این، شل را میتوان از طریق لاگین ریموت با استفاده از پروتکل متنی لاگین کرد. با این حال، این روش به خصوص از حیطه این کتاب فراتر است.

شل دیفالت در اکثر توزیع ها **Bash** می باشد، که بر پایه یک شل قدیمی تر به نام **Bourne Shell** است. شل های دیگری نیز در دسترس هستند؛ اکثرشان در کلیات با **Bash** تشابه دارند، با اینکه برخی از جزئیات متفاوت میباشند هر اکانت شل دیفالت خود را مشخص میکند، پس کاربران میتوانند شل های خود را به دلخواه تغییر دهند. (این کار با استفاده از ابزارهای مدیریت اکانت، مثل **usermod**، که در فصل 13 به آن پرداخته خواهد شد، صورت میگیرد).

نکته: شل های دیگر شامل **ksh**, **tcsh** و **zsh** میشوند. انتخاب شل یک موضوع سلیقه ای است. اگر تازه کار هستید، بهتر است با **Bash** شروع کنید صرفا چون محبوب تر است.

### Launching a terminal

اکثر توزیع های لینوکس به شما امکان نصب برنامه های ترمینال **ال-U** مختلف را میدهند. معمولاً، محیط های دسکتاپ با ترمینال های خودشان عرضه میشوند، پس گزینه های ترمینال شما بستگی به محیط دسکتاپ

شما دارد. بسیاری از ترمینال‌ها شامل کلمه ترمینال در نام خود می‌شوند، درحالی که برخی دیگر مثل برنامه ترمینال میتواند در هر محیط دسکتاپ متفاوت باشد. معمولاً میتوانید در منوهای محیط دسکتاپ خود یک ورودی پیدا کنید، همانطور که در فصل چهارم شرح داده شد. برای مثال، اگر از محیط دسکتاپ Cinnamon در لینوکس Mint استفاده می‌کنید، میتوانید ترمینال را در پنل پایین یا پنل Favorites پیدا کنید، یا این مراحل مربوط به منو را دنبال کنید:

1. بر روی منو کلیک کنید، معمولاً در گوشه پایین سمت چپ قرار دارند.
2. بر روی Administration کلیک کنید و از اسکرول برای دیدن گزینه‌های منو استفاده کنید. باید حداقل یک برنامه ترمینال مشابه با شکل 5.1 مشاهده کنید.
3. بر روی آیکون ترمینال کلید کنید، و ترمینال اجرا خواهد شد.



Figure 5.1 Reaching a terminal via a menu on Cinnamon

بسیاری از محیط‌های دسکتاپ یک روش سرچ برای برنامه‌های ترمینال فراهم می‌کنند. برای مثال، اگر شما از محیط دسکتاپ گنوم استفاده می‌کنید، مراحل ساده زیر را دنبال کنید:

1. بر روی Activities در گوشه چپ بالا کلیک کنید.
2. در سرچ باکس، term را سرچ کنید، همانطور که در شکل 5.2 نمایش داده شده است.
3. بر روی آیکون ترمینال از لیست بدست امده کلیک کنید و اجرا میشود.

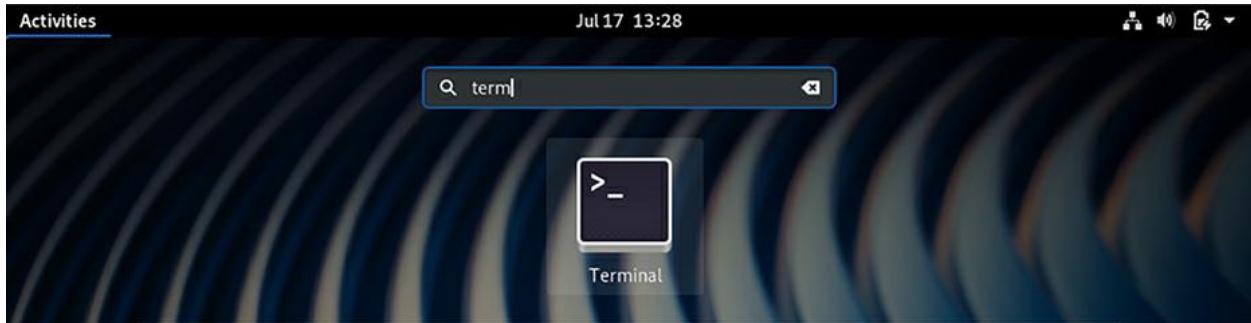


Figure 5.2 Reaching a terminal via a search on GNOME

پروسه سرچ برای برنامه های ترمینال میتواند در هر دسکتاپ شدیداً متفاوت باشد.

نکته: برخی توزیع ها به شما امکان باز کردن ترمینال با **Ctrl+Alt+T** را میدهند.

زمانی که اجرا می شود، ترمینال گنوم یک Prompt را نمایش میدهد؛ در شکل 5.3، مثال [~christine@localhost ~]\$ است. این مثال پرامپت دیفالت فدورا که شامل یوزرنیم شما، کامپیوترتان، دایرکتوری فعلی (علامت ~، نشانگر دایرکتوری home می باشد)، و یک علامت دلار میشود را نشان میدهد. برخی از این امکانات احتمال تغییر در حین استفاده از شل را دارند، همانطور که در فصل 7 توضیح داده میشود، اگر از توزیع دیگری استفاده میکنید، پرامپت احتمالاً در جزئیات تفاوت خواهد داشت، اگرچه اکثر پرامپت های دیفالت با علامت دلار (\$) یا علامت بزرگتر-کوچکتر (>) در شل های عادی تمام میشوند.



Figure 5.3 GNOME's Terminal program is typical and is dominated by a textual display area.

نکته: زمانی که از اکانت ادمینستور، `root`، استفاده میکنید، پرامپت معمولاً با علامت هشتگ (#) تمام میشود. فصل 12 اکانت `root` را با جزئیات بیشتری توصیف میکند.

اکثر ترمینال ها امکانات رایجی مثل برنامه های `GUI` را فراهم میکنند؛ میتوانید اندازه پنجره آنها را تغییر دهید، آنها را ببندید، گزینه هایی از منوها انتخاب کنید، و خیلی بیشتر. اگرچه جزئیات بستگی به برنامه ای که استفاده میکنید دارد. امکان دارد بخواهید گزینه های موجود در ترمینال خود را پیگیری کنید تا فونت دلخواهتان را تنظیم کنید، طرح رنگ برنامه را تغییر دهید و غیره.

بسیاری از برنامه های ترمینال از تب ها پشتیبانی میکنند، که مشابه تب های مرورگر وب هستند. در اکثر موارد، مثل ترمینال گنوم، میتوانید با کلیک بر + در سمت راست بالای ترمینال یک تب جدید باز کنید. باز کردن چندین تب می تواند کمک کننده باشد چون می توانید چندین برنامه را همزمان اجرا کنید، راحت تر در چندین دایرکتوری کار کنید، یا برنامه ها را همزمان به عنوان اکانت خودتان و `root` اجرا کنید. متناباً، میتوانید با اجرا کردن چندین برنامه ترمینال به نتایج مشابه برسید.

زمانی که کارتان با ترمینال تمام شد، میتوانید مثل هر برنامه دیگری آن را با کلیک بر روی `X` در گوش راست بالا ببندید، متناباً، میتوانید دستور `exit` را تایپ کنید و در پرامپت شل آن وارد کنید.

## Logging into text mode console

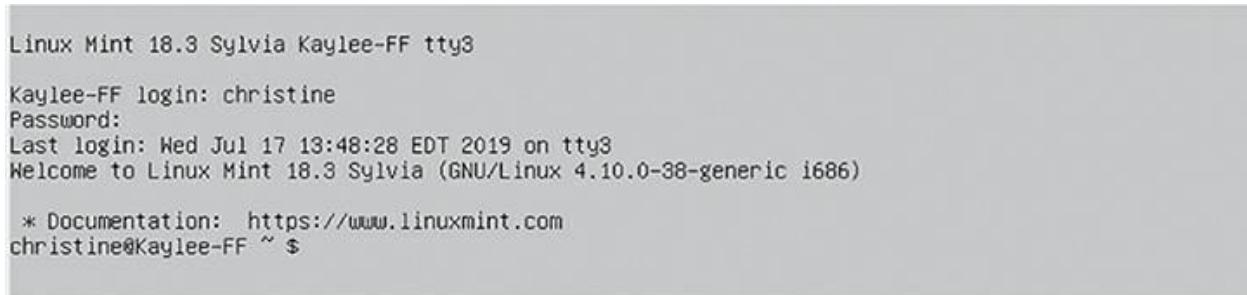
در نگاه اول، لینوکس از لحاظ سیستم عامل `GUI` بودن مشابه ویندوز و مکینتاش است. با این حال اگر به عمق بروید، یک اینترفیس متنی خالص پیدا خواهید کرد. لینوکس از ترمینال های مجازی (VTs) ) پشتیبانی میکند، که اسکرین های مجازی ای هستند که میتوانند انواع مختلف اطلاعات، متنی یا گرافیکی را در خود جای دهند. اکثر توزیع های لینوکس با شش یا هفت VTs اجرا میشنوند. در CentOS و Redhat، اولین VT معمولاً سیستم پنجره `GUI` را اجرا میکند. فدورا ورک استیشن، یک لاگین اسکرین گرافیکی را بر روی 1VT اجرا میکند، لاگین کردن در این اسکرین یک `GUI` بر روی 2VT اجرا میکند. بسیاری از توزیع های دیگر یک لاگین اسکرین گرافیکی بر روی 1VT فراهم میکنند و زمانی که کاربر لاگین میکند آن را با `GUI` جایگزین میکنند. دیگر توزیع ها از 7VT و 8VT برای لاگین اسکرین و `GUI` خود استفاده میکنند و 1VT را به عنوان نمایشگر متنی رها میکنند.

شما میتوانید بین VM ها با `Ctrl+Alt+Fn` سوئیچ کنید، که `Fn` یک کلید عملیاتی است. (حين سوئیچ کردن بین VT های متنی، `Alt+Fn` کافی است).

نکته: منظور از `Ctrl+Alt+Fn` نگه داشتن کلید `Ctrl`، نگه داشتن کلید `Alt` و فشار دادن کلید عملیاتی مورد نظر و رها کردن هر سه کلید به صورت همزمان است. کلید های عملیاتی (1F تا 12F) در بالا کیبورد قرار دارند. میتوانید عدد کلید عملیاتی ای که با عدد ترمینال همخوانی دارد را انتخاب کنید. برای مثال، برای 3VT از ترکیب `Ctrl+Alt+F3` استفاده کنید.

برای دسترسی و ورود به حالت متنی کنسول، گام های زیر را دنبال کنید:

1. ترکیب Ctrl+Alt+F3 را فشار دهید. یک پرامپت متنی مشابه با چند خط اول شکل 5.4 خواهد دید.
2. یوزرنیم خود را در این پرامپت لایپ کنید، و آن با پرامپت پسورد پاسخ خواهد داد. در شکل 5.4 یوزرنیم Christine است.
3. در پرامپت پسورد، رمز عبور خود را وارد کنید.
4. اگر اقدام به لاین شما موفقیت آمیز بود، یک پرامپت Bash مثل شکل 5.4 خواهد دید.



```

Linux Mint 18.3 Sylvia Kaylee-FF tty3
Kaylee-FF login: christine
Password:
Last login: Wed Jul 17 13:48:28 EDT 2019 on tty3
Welcome to Linux Mint 18.3 Sylvia (GNU/Linux 4.10.0-38-generic i686)

* Documentation: https://www.linuxmint.com
christine@Kaylee-FF ~ $

```

Figure 5.4 Reaching and logging into a VT

نکته: زمانی که پسورد خود را در پرامپت پسورد یک ترمینال کنسول مجازی وارد می کنید، هیچ چیزی نمایش داده نمیشود. نه نقطه ای و نه ستاره ای نمایش داده نخواهد شد از آنجایی که VT ها از یک لاین منیجر GUI استفاده میکنند.

شما میتوانید بین لاین متنی و GUI با استفاده از Ctrl+Alt+Fn پی در پی سوئیچ کنید. همچنین می توانید چندین لاین متنی ایجاد کنید و بین آنها به همان روش سوئیچ کنید. این قابلیت میتواند زمانی که سعی بر دیباگ کردن یک مشکل مربوط به GUI را دارید کارآمد باشد.

زمانی که کارتان با عملیات متنی خود تمام شد، دستور exit را تایپ کنید تا به آن خاتمه دهید. همچنین می توانید دستور logout را برای پایان عملیات خود وارد کنید.

## Running programs

پس از آن که یک ترمینال را باز کردید یا به ابزار متنی لاین کردید، وقت آن است که نحوه کار با Shell را یاد بگیرید. Shell شامل تعدادی دستور Built-in میباشد، اما اکثر کاری که در Shell انجام می دهید شامل اجرای برنامه های دیگر میشود. همانطور که در بخش های پیش رو توصیف خواهد شد، شما میتوانید برنامه های متنی و GUI را اجرا کنید. برخی اوقات ممکن است نیاز به اجرای یک برنامه در گراند را داشته باشید و استفاده از Shell را برای اهداف دیگر نگه دارید، که در خیلی شرایط ها میتواند مفید باشد.

# Understanding Text-Mode Program Syntax

ممکن است تصور کنید یک استاد برنامه نویسی تمام دستورهای مختلف را طراحی و خلق کرده است، اما چنین نیست. با اینکه برخی از دستورها برنامه نویس‌های مشترکی دارند، بسیاری از دستورها توسط افراد مختلف نوشته شده اند؛ در نتیجه روش استفاده از آنها هم مختلف است.

خوبی‌خانه، بسیاری از دستورات یک گرامر ساده پیروی می‌کنند:

نام دستور [آپشن] . . . [آرگومان]

در ساختار گرامر دستور:

- نام دستور: اسم دستور مورد استفاده برای اجرای برنامه
- [آپشن] آیتم‌های اضافی ای هستند که برای تغییر رفتار دستور استفاده می‌شوند. معمولاً انواع مختلفی از آپشن‌ها (به عنوان سوئیچ‌ها نیز شناخته می‌شوند) وجود دارد که می‌توانید به دستور اضافه کنید. برآکت ([]) به معنای اختیاری بودن آپشن‌ها می‌باشد، و سه نقطه (...) نشان دهنده این است که شما می‌توانید بیش از یک آپشن استفاده کنید.
- [آرگومان] آیتمی است که به دستور می‌دهید تا به آن بفهمانید میخواهید روی این آیتم اجرا شود. یک آرگومان همچنین می‌تواند یک دستور ثانویه (دستوری که بخشی از یک دستور بزرگتر است) باشد. همانطور که مشاهده می‌کنید این مورد هم به دلیل برآکت‌ها اختیاری است و همچنین می‌توانید چندین آرگومان را به برنامه بدهید.

نکته: اگر میخواهید بیش از یک آپشن دستور استفاده کنید، می‌توانید آنها را به یکدیگر بچسبانید. برای مثال، برای استفاده از آپشن‌های `a` و `b`- `who -ab` می‌نویسید

برنامه `who` به ما نشان میدهد کدام کاربران در آن لحظه در سیستم هستند:

```
$ who
rich          tty7        2020-07-16 16:40 (:0)
christine    pts/1      2020-07-16 16:37 (192.168.0.102)
```

نکته: در این کتاب، دستوراتی شما وارد می‌کنید با فونت `bold` و خروجی دستور با فونت `monospace` نوشته شده اند.

با این حال، می‌توانید خروجی نمایشی این دستور را با اضافه کردن آپشن `b`- به آن تغییر دهید. در این مورد، برنامه زمان استارت سیستم را نشان میدهد:

```
$ who -b
system boot 2020-07-16 16:16
```

نمونه ای از یک دستور که آرگومان ها را قبول میکند دستور `cat` است، که مخفف کلمه `concatenate` می باشد. دستور `cat` میتواند به سرعت یک فایل متنی کوتاه را نمایش دهد. در این مثال، دستور `cat` نام فایل `MyFile.txt` را به عنوان یک آرگومان میگیرد:

```
$ cat MyFile.txt  
This is the content of MyFile.txt
```

نکته: دستورها، آرگومان ها و نام فایل ها در شل به حروف بزرگ و کوچک حساس هستند.

دقت کنید که آپشن ها (سوئیچ ها) بین دستورهای مختلف استاندارد نشده اند. برای مثال، زمانی که از سوئیچ `-b`- با برنامه `cat` استفاده میکنید، اطلاعاتی راجع به استارتاتپ سیستم مثل موقعي که با دستور `who` انجام می داد به شما نشان نمیدهد، در عوض شماره خط به خطوط غیر خالی متن فایل اضافه میکند:

```
$ cat -b MyFile.txt  
1 This is the contents of MyFile.txt
```

میتوانید در رابطه با جزئیات استفاده از دستورات از طریق Linux Man Pages اطلاعات کسب کنید. برنامه مورد استفاده برای انجام این کار `man` نام دارد، و می توانید یک نام دستور را که میخواهید به عنوان آرگومان آن را یاد بگیرید به آن بدهید، مثل `man cat` برای یاد گرفتن راجع به دستور.

نکته: بخش "راهنمای استفاده از Man Page ها" در این فصل، سیستم `Man page` و دیگر مطالب را با جزئیات بیشتر توضیح میدهد.

متاسفانه، بسیاری از دستورات از گرامر استاندارد دستور پیروی نمیکنند. اما نامید نشوید. پیج های راهنمای برای سیستم لینوکس ساختار گرامر بسیاری از دستورات را به شما میدهد. برای پیدا کردن ساختار گرامر یک دستور بخصوص، `man` آن را بررسی و بخش `Synopsis` آن را چک کنید.

دستور `man` ویزگی برخی برنامه های متنی را شرح میدهد: آن ها میتوانند زمان اجرا شدن کنترل تمام ترمینال را به دست بگیرند. در مورد `man`، میتوانید با کلید های فلش در کیبورد (بالا و پایین) در مطالب آن اسکرول کنید. ادیتور های متنی، مثل `emacs`، `vi`، `nano` از امکانات مشابهی استفاده میکنند.

## Running Text-Mode Programs

لینوکس برنامه ها را در لوکیشن های مختلفی ذخیره میکند، مثل `/bin` و `/usr/bin` و `/sbin` و `/usr/local/bin`. (برنامه هایی که عموماً توسط `root` مورد استفاده قرار میگیرند در `/` یا `/usr/local/sbin` و یا `/usr/sbin` نمیگیرند.) اگر یک برنامه قابل اجرا در یکی از این دایرکتوری ها ظاهر شود (که شامل `path` میشوند)، می توانید آن را به سادگی با تایپ کردن اسمش آن را اجرا کنید:

```
$ free
```

	total	used	free	shared	buffers	cached
Mem:	3798016	3759004	39012	0	24800	1117444
-/+ buffers/cache:	2616760	1181256				
Swap:	6291452	0	6291452			

این دستور نمونه اطلاعاتی در رابطه با مصرف کامپیوتر از memory را نشان میدهد. البته شما نیازی به نگرانی در رابطه با جزئیات خروجی این دستور ندارید؛ فقط به خاطر بسپارید که برنامه free اطلاعات را در همان ترمینالی که از آن استفاده شد نمایش داد. در فصل 9 برنامه free با جزئیات بیشتری پوشش داده میشود.

شما با تایپ کردن دستور زیر میتوانید بفهمید چه دایرکتوری هایی در *path* یا *defined path* قرار دارند:

```
$ echo $PATH
```

نتیجه یک ستون از نام دایرکتوری ها خواهد بود، که شل به ترتیب آن ها را زمانی که یک دستور خاص وارد میکنید که به صورت مستقیم توسط شل مدیریت نمیشود، جستجو میکند. *PATH* یک متغیر محیطی است. متغیر های محیطی با جزئیات بیشتر در فصل 9 پوشش داده شدند.

شما میتوانید یک برنامه را که جزئی از دایرکتوری های *PATH* نیست با نوشتن مسیر دایرکتوری برنامه همراه با نام آن اجرا کنید. برای مثال:

```
$ /home/christine/myprogram
```

نکته: زمانی که یک برنامه قابل اجرا که در *PATH* قرار ندارد را اجرا میکنید، این کار احضار یک دستور خارج از مسیر معین شده نام دارد. (Invoking a command outside the defined path)

اگر مایلید مشخص کنید چگونه یک برنامه قابل اجرا هندل می شود، میتوانید از دستور type استفاده کنید. برای مثال:

```
$ type free
free is /usr/bin/free
```

نتیجه به شما مسیر دایرکتوری برنامه را نشان میدهد (که ممکن است با مسیر نشان داده شده در مثال بسته به نوع توزیع لینوکس متفاوت باشد). اگر متغیر محیطی *PATH* شامل مسیر دایرکتوری شود، شما تنها نیاز به وارد کردن نام برنامه برای اجرای آن دارید.

## Running GUI Programs

شما میتوانید برنامه های GUI را از یک ترمینال مثل برنامه های متنی اجرا کنید (اگرچه در صورتی که توسط یک VT متنی لاگین کردید این کار ممکن نیست). شما باید نام فایل را برای اجرا بدانید. نام فایل معمولاً به نامی که برای اجرای برنامه ها از منوهای دسکتاپ استفاده میکنید مرتبط است، اما لزوماً یکسان نیست. برای مثال، نام فایل مرورگر فایرفاکس صرفاً ( بدون پسوند firefox ) می باشد. پس برای اجرا آن نیاز به تایپ صرفاً firefox را دارد.

برخی برنامه های GUI خروجی های متنی ای تولید میکنند که برای پیدا کردن منشا مشکلات مفید هستند، پس اجرای یک برنامه از ترمینال میتواند قدم اول مناسبی برای دیباگ کردن مشکلات باشد. شما همچنین بهتر است برنامه ها را با این روش اجرا کنید چون سریعتر از گشتن برای آیکون برنامه ها در منوهای دسکتاپ است یا چون گاهی برخی برنامه ها در دسکتاپ ظاهر نمیشوند.

## Running Programs in the Background

زمانی که یک برنامه GUI را از پنجره یک ترمینال اجرا میکنید، برنامه GUI پنجره یا پنجره های خاص خود را باز میکند. پنجره ترمینال باز خواهد ماند اما معمولاً بی واکنش خواهد ماند. اگر میخواهید دستورهای دیگری در همین پنجره وارد کنید، میتوانید با فشار دادن ترکیب Ctrl+Z این کار را انجام دهید. این کار باعث تعليق برنامه در حال اجرا میشود - به این معنا که برنامه به خواب میرود. در این حالت تعليق، برنامه GUI پاسخی به ورودی ها یا هرگونه دستوری نخواهد داد. با اين حال، اکنون میتوانید از پنجره ترمینال برای وارد کردن دستورات استفاده کنید.

پس از آنکه یک برنامه را به تعليق در آوردید، اگر میخواهید برنامه و ترمینال باهم همزمان اجرا شوند، میتوانید از دستور `bg` ( مخفف Background ) استفاده کنید. اکنون هر دو برنامه فعال خواهند بود.

اگر میخواهید فقط برنامه تعليقی را فعال کنید، دستور `fg` را تایپ کنید. اين دستور برنامه GUI تعليقی را به حالت Foreground باز میگردد، و باري دیگر ترمینال را غيرفعال میکند. به خاطر بسیارید که در این زمینه، اصطلاحات Foreground و Background اشاره ای به رابطه برنامه با شل دارد، نه به موقعیت پنجره برنامه در جمعی از پنجره های روی اسکرین.

نکته: فشار دادن Ctrl+Z همچنین باعث تعليق برنامه های متنی نيز میشود که به شما امكان استفاده از شل قبل از بازگشت به برنامه با دستور `fg` را ميدهد.

اگر قبل از اجرا میدانید که میخواهید برنامه در بک گراند اجرا شود، میتوانید با اضافه کردن یک علامت امپرسند (&) به آخر دستور اين کار را مانند مثال زير انجام دهید:

```
$ firefox&
```

همانطور که در شکل 5.5 نمایش داده شده، این دستور فایرفاکس را در بک گراند اجرا میکند و به شما امکان استفاده از فایرفاکس و ادامه استفاده از شل را میدهد.

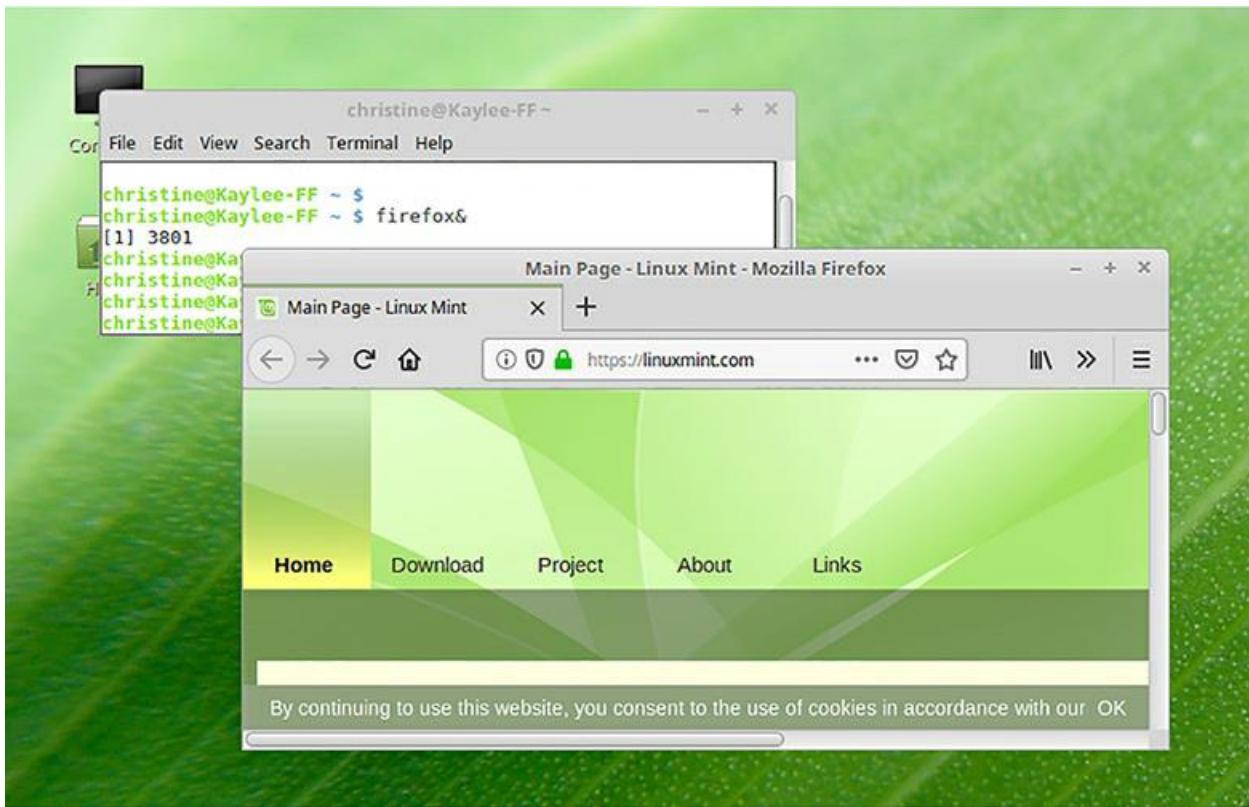


Figure 5.5 Launching Firefox in the background to allow use of both the web browser and the shell

این ویژگی بک گراند برای اجرای برنامه های GUI مفیدترین است، اما برخی اوقات برای برنامه های متنی نیز به کار میرود. یک برنامه پیچیده فشرده سازی اعداد میتواند به طوری طراحی شود که از چند دقیقه تا چندین ساعت بدون خروجی کار کند. درنتیجه بهتر است در بک گراند اجرا شود تا در این مدت کنترل شل خود را داشته باشید. با این حال، بهتر است بدانید که اگر یک برنامه را در بک گراند اجرا کنید که خروجی خود را در شل ثبت میکند، آن خروجی در ترمینال شما ظاهر میشود و احتمالاً در کاری که در آن لحظه انجام میدهید مداخله میکند.

## Using Shell Features

Bash شامل چندین امکانات مختلف می شود که استفاده از آن را راحت تر میکند. برخی از آنها تا اینجا توصیف شدند. بسیاری دیگر از حیطه این کتاب خارج اند. با این حال، دو مورد دیگر لایق توجه اند: تکمیل دستور و تاریخچه دستور.

## Using command completion

تکمیل دستور یک قهرمان برای کسانی که از تایپ کردن متنفرند می باشد: این یک راهی است برای وارد کردن یک دستور یا نام فایل بلند با حداقل استفاده از دکمه های کیبورد. برای استفاده از تکمیل دستور، شما بخشی از دستور یا نام فایل را تایپ میکنید و سپس کلید Tab را فشار میدهید. اگر مقدار حروف برای کامل کردن کافی باشد Bash کل دستور یا نام فایل را تکمیل میکند.

نکته: اگر تکمیل دستور برای شما کار نمیکند، احتمالا برنامه ترمینال ویژگی آن را خاموش کرده است.

برای امتحان استفاده از تکمیل دستور، میتوانید مراحل زیر را امتحان کنید:

1. شل را اجرا کنید

2. حروف wh را تایپ کنید و کلید Tab را فشار دهید. ممکن است کامپیوتر با یک صدای beep یا صدای دیگر واکنش دهد. این یعنی دستور ناقص شما قابل تکمیل شدن است و میتواند به چندین دستور تکمیل شود، پس شما باید حروف بیشتری تایپ کنید. (در برخی پیکربندی ها، کامپیوتر مستقیماً به مرحله بعد میرود انگار شما دوبار کلید tab را فشار دادید)

3. کلید tab را دوباره فشار دهید. شل یک لیستی از دستورهای محتمل مثل whatis و whereis را نشان دهد.

4. حروف oa را تایپ کنید، تا اینجا دستور شما whoa میشود. سپس کلید tab را فشار دهید. کامپیوتر دستور whoami را تکمیل خواهد کرد. (اگر تکمیل نشد، یعنی دستور دیگری با حروف اولیه مشابه وجود دارد و شما نیاز به تایپ چند حرف بیشتر دارید).

5. کلید enter را فشار دهید. کامپیوتر دستور whoami را اجرا میکند، که نام اکانتی که در حال حاضر استفاده میکنید را نشان میدهد.

گاهی اوقات تکمیل دستور میتواند یک دستور را فقط بخش به بخش کامل کند. برای مثال، تایپ کردن grub فقط حرف b را اضافه میکند. با این حال، چندین دستور با grub آغاز میشوند، پس بهتر است خودتان چند حروف بیشتر اضافه کنید. (این دستورات با GRUB Grand Unified Bootloader یا لینوکس کمک میکند کار دارند).

نکته: برخی جزئیات از چگونگی عملکرد تکمیل دستور در هر توزیع متفاوت است.

تکمیل دستور همچنین با فایل ها نیز کار میکند. برای مثال، شما میتوانید cat /etc/ser را به همراه کلید tab تایپ کنید تا bash نام فایل و در نتیجه دستور را مثل cat /etc/services کامل کند. (این دستور به شما محتویات یک فایل پیکربندی لینوکس را نشان میدهد).

## Using command history

Bash دستورات اخیری که تایپ کرده اید را به خاطر میسپارد، و شما میتوانید از این موضوع برای صرفه جویی در زمان استفاده کنید. اگر نیاز به تایپ دستوری دارید که به یکی از دستورات اخیر شباهت داشته باشد، به عنوان ساده ترین روش میتوانید از کلید فلش بالا استفاده کنید تا دستور قبلی را وارد کنید؛ فشار دادن کلید

فلش بالا به طور مداوم به دستورات قبلی حرکت می‌کند. جدول 5.1 خلاصه‌ای از برخی از دستورات کلیدی متداول است که می‌توانید در تاریخچه دستورات یا حتی در ویرایش دستورات جدید استفاده کنید.

Table 5.1 Bash editing and command history features

Keystroke	Effect
Up arrow	بازیابی ورودی قبلی از تاریخچه فرمان
Down arrow	بازیابی یک ورودی قبلی که با استفاده از فلش بالا نادیده گرفته شده
Left arrow	حرکت مکان‌نما به سمت چپ یک کاراکتر
Right arrow	حرکت مکان‌نما به سمت راست یک کاراکتر
Ctrl+A	حرکت مکان‌نما به ابتدای خط
Ctrl+E	حرکت مکان‌نما به انتهای خط
Delete key	حذف کاراکتر زیر مکان‌نما
Backspace Key	حذف کاراکتر به سمت چپ مکان‌نما
Ctrl+T	تبادل کاراکتر زیر مکان‌نما با کاراکتر سمت چپ آن
Ctrl+X and Then Ctrl+E	راهاندازی یک ویرایشگر کامل بر روی کامند لاین فعلی
Ctrl+R	جستجوی یک فرمان. چند کاراکتر تایپ کنید و شل آخرین فرمانی را که شامل آن کاراکترها باشد پیدا می‌کند. می‌توانید با فشردن Ctrl+R دوباره برای جستجوی فرمان قبلی که شامل آن کاراکترها باشد، جستجو کنید.

نکته: بسیاری از ویژگی‌های ویرایش دستورات در Bash مشابه با ویژگی‌های استفاده شده توسط ادیتور متن emacs هستند.

به عنوان یک مثال از استفاده از تاریخچه دستورات، این را امتحان کنید:

1. دستور `echo $PATH` را تایپ کرده و کلید Enter را فشار دهید تا دایرکتوری‌هایی که مسیر تعریف شده شما را تشکیل می‌دهند، مشاهده کنید.

2. کلید فلش بالا را فشار دهید. دستور `echo $PATH` شما باید دوباره ظاهر شود.

3. پنج بار کلید Backspace را فشار دهید تا `$PATH` را حذف کنید.

4. دستور `Hello World` را تایپ کنید تا دستور جدید `echo Hello World` شود و سپس Enter را فشار دهید. حالا باید کلمات `Hello World` را روی اسکرین خود ببینید.

5. کلید Ctrl+R را فشار دهید. Bash prompt تغییر خواهد کرد و حالا به این صورت است:

(reverse-i-search) `^:.

6. P را تایپ کنید (بدون فشار دادن Enter). دستور قبلی `echo $PATH` شما ظاهر می‌شود.

7. Enter را فشار دهید. دستور `echo $PATH` باید دوباره اجرا شود و پaramپت شل Bash شما باید به حالت عادی بازگردد.

ویژگی جستجو با استفاده از Ctrl+R هر چیزی که در یک کامند لاین وارد می‌کنید جستجو می‌کند، مثل نام دستور، نام فایل یا سایر پارامترهای دستور.

یک ویژگی دیگر از تاریخچه، دستور `history` است. دستور `history` را تایپ کنید تا تمام دستورات موجود در تاریخچه شما را مشاهده کنید، یا یک عدد (مانند 10 `history 10`) اضافه کنید تا 10 دستور اخیر خود را مشاهده کنید. در ادامه همراه با دستورات، یک شماره خواهید دید که می‌توانید از این شماره برای اجرای یک دستور از تاریخچه‌تان استفاده کنید. فقط کافیست ! را تایپ کرده و پس از آن شماره دستور را اضافه کنید و Enter را فشار دهید و شل دستور انتخابی شما را همراه با نتایج دستور نمایش می‌دهد.

بهتر است ویژگی‌ها تجربه کنید. تکمیل و تاریخچه دستورات هر دو ابزار قدرتمندی هستند که می‌توانند به شما کمک کنند تا از تایپ تکراری زیادی جلوگیری کنید. تاریخچه دستورات همچنین می‌تواند به عنوان یک وسیله حافظه مفید باشد. به عنوان مثال، اگر اسم دقیق یک فایل یا دستور را که اخیراً استفاده کرده‌اید را فراموش کرده‌اید، ممکن است با جستجو بر روی بخشی از نامی که به یاد دارید، آن را بازیابی کنید.

## Getting Help Using Man Pages

گاهی اوقات برای پادآوری آرگومان‌ها یا گزینه‌هایی که یک دستور می‌تواند استفاده کند، یا نحوه صحیح وارد کردن یک دستور نیاز به کمک دارید. صفحات راهنما (همچنین به آنها صفحات *man* گفته می‌شود) می‌توانند کمک کنند. صفحات راهنما نه تنها برنامه‌ها را توضیح می‌دهند بلکه فایل‌های پیکربندی و ویژگی‌های دیگر نصب لینوکس را نیز توضیح می‌دهند.

قبل از استفاده از آنها، باید با اهداف آنها و از این روی با قابلیت‌ها و محدودیت‌های آنها آشنا شوید. با در نظر گرفتن این اطلاعات، می‌توانید به دنبال کمک در سیستم صفحات *man* بگردید. از جمله جستجو برای صفحات *man* بر اساس بخش یا استفاده از ابزار *what is* یا *apropos* برای جستجو کلمات کلیدی. هنگامی که یک صفحه *man* را می‌خوانید، آشنایی با ساختار آن می‌تواند به شما کمک کند تا به سرعت به اطلاعاتی که نیاز دارید دست یابید.

## Understanding the Purpose of Man Pages

صفحات *man* لینوکس می‌توانند یک منبع بسیار مفید باشند، اما برخلاف سیستم‌های راهنما در برخی سیستم‌عامل‌ها، صفحات *man* لینوکس قرار نیست طبقه‌بندی شوند و جنبه آموزشی داشته باشند. آنها به عنوان مرجع سریع برای کمک به کسی که حداقل تا حدی با یک دستور، فایل پیکربندی یا دیگر امکانات سیستم عامل آشنا است، طراحی شده‌اند. آنها بیشترین کارایی را زمانی دارند که نیاز به دانستن نام یک گزینه در یک فایل پیکربندی، یا آپشن‌هایی که باید با یک دستور استفاده کرد و یا جزئیات مشابه دارید. اگر نیاز به یادگیری یک برنامه جدید از ابتدا دارید، معمولاً داکیومنت‌های دیگر انتخاب بهتری هستند.

صفحات راهنما همچنین تفاوت‌های زیادی در کیفیت دارند؛ برخی از آنها بسیار خوب هستند، اما برخی دیگر به شدت مختصر و گاهی حتی نادرست هستند. در بیشتر موارد، برنامه‌نویسانی که برنامه مربوطه را نوشته‌اند، آنها را نوشته‌اند.

نکته: بخش پیش رو "Finding Additional Documentation" توضیح می‌دهد که چگونه می‌توان مستنداتی را که آموزشی‌تر از صفحات *man* هستند، پیدا کرد.

در این کتاب، بسیاری از دستورات لینوکس به شیوه آموزشی توضیح داده شده‌اند. با این حال، گاهی اوقات اطلاعاتی در مورد آپشن‌های پنهان، اثرات ناشناخته برنامه، و موارد مشابه حذف می‌شود. اصولاً، صفحات *man* باید این نکات دقیق را پوشش دهند. این امر صفحات *man* را در صورتی که نیاز به اطلاعات بیشتری داشته باشید به یک منبع عالی برای یادگیری بیشتر در مورد دستورات مورد توضیح در این کتاب تبدیل می‌کند.

## Locating Man Pages by Section Number

در ساده‌ترین حالت، می‌توانید یک صفحه man را با تایپ کردن `man` و در ادامه نام یک دستور، فایل پیکربندی، صدا زدن سیستم، یا کلمه کلیدی دیگری مشاهده کنید. هر صفحه man به یکی از نه دسته اصلی تقسیم می‌شود، که در جدول 5.2 خلاصه شده است.

Table 5.2 Manual Sections

Section number	Description
1	برنامه‌ها و دستورات شل را اجرا می‌کند
2	کال‌های سیستم فراهم شده توسط کرنل
3	کال‌های کتابخانه فراهم شده توسط کتابخانه کرنل
4	فایل‌های دستگاه (معمولًا در <code>/dev</code> ذخیره شده‌اند)
5	فرمت‌های فایل‌ها
6	بازی‌ها
7	متفرقه (پکیج‌های ماکرو، قراردادها و غیره)
8	دستورات مدیریتی سیستم (برنامه‌هایی که اکثراً یا انحصاراً توسط <code>root</code> اجرا می‌شوند)
9	روتین‌های کرنل

اگر از کلمات کلیدی با دستور `man` استفاده کنید، باید آگاه باشید که ممکن است به ورودی‌هایی در چندین بخش منتهی شود. در چنین مواردی، ابزار `man` ورودی را در بخش مرتبط با ترتیب جستجو که به طور معمول توسط تنظیم SECTION در فایل پیکربندی `/etc/man_db.conf` یا فایل پیکربندی `/etc/manpath.config` (بسته به توزیع شما) مشخص شده است، باز می‌گرداند.

نکته: ترتیب دقیق جستجو در صفحات man که توسط SECTION مشخص شده، از یک توزیع به توزیع دیگر متفاوت است، اما معمولاً بخش 1 به عنوان اولین بخش جستجو می‌شود، سپس بخش 8 و سپس بخش‌های دیگر.

می‌توانید این رفتار جستجوی پیش‌فرض را با ارسال یک شماره بخش قبل از کلمه کلیدی لغو کنید. به عنوان مثال، تایپ کردن `man passwd` اطلاعاتی از بخش 1 درباره دستور `passwd` را باز می‌گرداند، اما تایپ کردن `man 5 passwd` اطلاعاتی از بخش 5 درباره فرمت فایل `/etc/passwd` را باز می‌گرداند. برخی از صفحات `man` ورودی‌هایی در بخش‌های با شماره‌های متغیر دارند که شامل پسوند `p` هستند، مانند بخش `p1`. اینها به صفحات `man` استاندارد واسط سیستم عامل پرتابل (POSIX) اشاره دارند، برخلاف صفحات `man` لینوکس که اغلب توسط افرادی نوشته شده‌اند که برنامه‌های لینوکس متن‌باز را که صفحات `man` آنها را توصیف می‌کنند، نوشته‌اند.

اگر تازه با لینوکس آشنا شده‌اید، احتمالاً بیشتر علاقه‌مند به بخش 1 (برنامه‌ها و دستورات قابل اجرا) خواهد بود، که معمولاً اولین بخش در ترتیب جستجوی صفحات `man` است - با این حال بخش 6، بازی‌ها، نیز ممکن است جالب باشد اگر زمان کافی داشته باشد! با پیشروی در وظایف پیشرفته و مدیریتی، بخش‌های 4، 5 و 8 برای فایل‌های دستگاه، فرمت‌های فایل و برنامه‌های اجرا شده توسط `root` مهم هستند. بخش‌های 2، 3 و 9 که فراخوان‌ها و روتین‌های کرنل را پوشش می‌دهند، بیشتر به برنامه‌نویسان جذابیت دارند.

## Searching for a Man Page

یکی از مشکلات مربوط به صفحات `man` این است که یافتن کمک در مورد یک موضوع ممکن است دشوار باشد مگر اینکه نام دقیق دستور، فراخوانی سیستم، یا فایلی که می‌خواهید استفاده کنید را بدانید. خوب‌بختانه، روش‌هایی برای جستجو در پایگاه داده راهنمای وجود دارند و می‌توانند به شما کمک کنند تا به یک صفحه `man` مناسب برسید:

**جستجوی خلاصه:** دستور `whatis` اطلاعات خلاصه موجود در صفحات `man` را برای کلمه کلیدی که مشخص می‌کنید جستجو می‌کند. این دستور یک خط خلاصه برای هر صفحه `man` مطابق با جستجو به شما برمی‌گرداند. (این خلاصه بخش نام است که در "خواندن صفحات `Man`" به زودی توضیح داده خواهد شد.) سپس می‌توانید از این اطلاعات برای یافتن و خواندن صفحه `man` مورد نیاز استفاده کنید. این دستور بیشترین کارایی را برای یافتن همه صفحات `man` در یک موضوع دارد. به عنوان مثال، تایپ کردن `whatis passwd` خطوطی را برمی‌گرداند که وجود ورودهای صفحه `man` برای `passwd` در بخش‌های مختلف را تأیید می‌کند.

**جستجوی دقیق:** دستور `apropos` یک جستجوی دقیق‌تری هم‌زمان در بخش‌های نام و توضیح صفحات `man` انجام می‌دهد. نتیجه بسیار شبیه به نتایج یک جستجوی `whatis` است، با این تفاوت که احتمالاً حاوی تعداد زیادی نتیجه خواهد بود. در واقع، انجام یک جستجوی `apropos` بر روی یک کلمه کلیدی بسیار متداول، مانند "the"، احتمالاً باعث بازگشت تعداد زیادی نتیجه می‌شود و جستجو بی‌فایده خواهد بود. جستجو بر روی یک کلمه کمتر متداول ممکن است مفیدتر باشد. به عنوان مثال، تایپ کردن `apropos`

ممکن است 16 ورود را برگرداند که شامل ورودهایی برای `passwd`, `smbpasswd`, `gpasswd` باشد که همگی ابزارها یا ابزارهای مرتبط با فایل‌های رمزبور هستند.

**نکته: آپشن `-k`** - در دستور `man` معادل دستور `apropos` است. بنابراین، می‌توانید یا `apropos` را تایپ کنید یا `man -k keyword` را تایپ کنید.

اگر از هر دو دستور `whatis` یا `apropos` پاسخ "nothing appropriate" دریافت کنید، معمولاً باید کلمه کلیدی که استفاده می‌کنید را تغییر دهید. با این حال، این ممکن است نشان دهد که پایگاه داده `man` به روز نشده است. این معمولاً در یک نصب تازه لینوکس یا پس از نصب یک برنامه جدید رخ می‌دهد. می‌توانید پایگاه داده `man` را به صورت دستی با استفاده از `Privilege` مدیر سیستم به روز کنید و دستور `makewhatis` (در توزیع‌های لینوکس قدیمی‌تر) یا `mandb` را تایپ کنید. (استفاده از `Privilege` مدیر سیستم در فصل 12 توضیح داده شده است).

**نکته: جزئیات در مورد اینکه چه صفحات `man` ای در دسترس هستند، از یک توزیع به توزیع دیگر متغیر است. این بر نتایج هر دو جستجوی `whatis` و `apropos` تاثیر خواهد داشت.**

## Reading Man Pages

کنوانسیون صفحات `man` یک سبک مختصر است که از چند بخش استفاده می‌کند، هر کدام با هدف خاصی. این نوع سازمان‌دهی طراحی شده است تا به شما در یافتن اطلاعات مورد نیاز کمک کند. به عنوان مثال، اگر مطلبی را جستجو کرده باشید که می‌دانید در یک بخش خاص است، می‌توانید به سرعت به آن بخش منتقل شویید. بخش‌های معمول شامل موارد زیر هستند:

**نام:** یک صفحه `man` با یک اظهارنامه از دستور، فراخوانی یا فایلی که توصیف می‌شود، همراه با چند کلمه توضیح آغاز می‌شود. به عنوان مثال، صفحه `man` برای `man` (بخش 1) یک بخش "نام" دارد که `interface to the on-line reference manuals` اجرا می‌کند.

**خلاصه:** توضیح کوتاهی از چگونگی استفاده از یک دستور را فراهم می‌کند. پارامترهای اختیاری در برآکت ظاهر می‌شوند، مانند `[D-]`. نقطه چین (...). یک مجموعه اختیاری از عناصر تکرار شونده را نشان می‌دهد، مانند چندین نام فایل، اگر یک دستور یک یا چند نام فایل را به عنوان گزینه‌ها بپذیرد. برخی از دستورات چندین خط خلاصه ارائه می‌دهند، که نشان دهنده این است که برخی از آپشن‌ها به بعضی دیگری وابسته هستند.

**توضیحات:** توضیحات یک خلاصه به زبان انگلیسی از کاربرد دستور، فایل یا عنصر دیگر است. توضیحات می‌تواند از یک خلاصه کوتاه تا چندین صفحه طول داشته باشد.

**آپشن:** این بخش بر آپشن‌هایی که در بخش خلاصه آورده شده است، توضیحات بیشتری ارائه می‌دهد. به طور معمول، هر آپشن در یک لیست ظاهر می‌شود با توضیحات یک پاراگرافی را زیر هر آپشن.

**فایل‌ها**: این بخش فهرستی از فایل‌هایی را که با موضوع صفحه man مرتبط هستند، ارائه می‌دهد. این فایل‌ها ممکن است فایل‌های پیکربندی برای یک سرور یا برنامه دیگر، فایل‌های پیکربندی مرتبط، فایل‌هایی که توسط موضوع صفحه تغییر می‌کنند و غیره باشند.

**See Also**: این بخش نشانگرهایی به اطلاعات مرتبط در سیستم man فراهم می‌کند، معمولاً با یک شماره بخش پیوسته. به عنوان مثال، صفحه man برای whatis به صفحات man برای apropos و چند ابزار مرتبط دیگر ارجاع دارد.

**باگ‌ها**: بسیاری از صفحات man یک بخش اشکالات دارند که نویسنده در آن اشکالات یا محدودیت‌های شناخته‌شده را توضیح می‌دهد یا اعلام می‌کند که هیچ اشکال شناخته‌شده‌ای وجود ندارد.

**تاریخچه**: برخی از صفحات man خلاصه‌ای از تاریخچه برنامه را فراهم می‌کنند، با ذکر تاریخ شروع پروژه و دستاوردهای اصلی بین آن زمان و نسخه فعلی. این تاریخچه به هیچ وجه در حد تغییرات فایل که با بیشتر سورس کد برنامه‌ها همراه است، جامع نیست.

ممکن است صفحات دستور خاص دارای بخش‌های کمتر، بیشتر یا با بخش‌های متفاوتی نسبت به اینها باشند. به عنوان مثال، بخش خلاصه اغلب از صفحات man مربوط به فایل‌های پیکربندی حذف می‌شود. صفحات man با توضیحات بسیار طولانی اغلب بخش توضیحات را به چند بخش تقسیم می‌کنند، هر کدام با عنوان خود.

شکل 5.6 یک صفحه man نمونه در یک پنجره ترمینال نشان می‌دهد. همانطور که مشاهده می‌شود، نام‌های بخش با حروف بزرگ و پرنگ نمایش داده می‌شوند که به شما کمک می‌کنند بخش‌های مرتبط را به راحتی در طول مستندات پیدا کنید.

```

christine@Kaylee-FF ~
File Edit View Search Terminal Help
WHATIS(1) Manual pager utils WHATIS(1)
NAME
whatis - display one-line manual page descriptions

SYNOPSIS
whatis [-dlv?V] [-r|-w] [-s list] [-m system[,...]] [-M path] [-L
locale] [-c file] name ...

DESCRIPTION
Each manual page has a short description available within it. whatis
searches the manual page names and displays the manual page descrip-
tions of any name matched.

_name_ may contain wildcards (-w) or be a regular expression (-r). Using
these options, it may be necessary to quote the name or escape (\) the
special characters to stop the shell from interpreting them.

index databases are used during the search, and are updated by the
mandb program. Depending on your installation, this may be run by a
periodic cron job, or may need to be run manually after new manual
pages have been installed. To produce an old style text whatis data-
base from the relative index database, issue the command:

```

Manual page whatis(1) line 1 (press h for help or q to quit)

Figure 5.6 The formatting of man pages helps you locate information quickly.

نکته: به صورت پیشفرض، دستور `man` از برنامه `less` استفاده میکند تا به شما امکان حرکت به جلو و عقب در مستندات استفاده را بدهد، و زمانی که کارتان تمام شد، از صفحات `man` خارج شوید. بخش بعدی به نام "استفاده از `less`" جزئیات این برنامه را توضیح می‌دهد.

## Using less

سیستم `man` لینوکس از یک برنامه به نام `less` برای نمایش اطلاعات استفاده می‌کند. این برنامه یک `pager` است که یک فایل متنی را یک صفحه در زمان نمایش می‌دهد. شما می‌توانید به جلو یا به عقب در فایل حرکت کنید، به یک خط خاص بروید و برای یافتن اطلاعات جستجو کنید. جدول 5.3 خلاصه‌ای از راههای متداول حرکت در یک سند با استفاده از `less` را ذکر می‌کند.

نکته: یک `pager` قدیمی‌تر با نام `more` وجود داشت، اما `less` ویژگی‌های بیشتری افزود. این نام عجیب نمونه‌ای از شوخی گیکی است.

Table 5.3 `less` file-navigation commands

Keystroke	Action
H	نمایش راهنمای استفاده از <code>less</code>
Page Down, spacebar, Ctrl+V, F, or Ctrl+F	حرکت یک صفحه به پایین در سند
Page Up, Esc+V, B, or Ctrl+B	حرکت یک صفحه به بالا در سند
Down arrow, Enter, Ctrl+N, E, Ctrl+E, j, or Ctrl+J	حرکت یک خط به پایین در سند
Up arrow, Y, Ctrl+Y, Ctrl+P, K, or Ctrl+K	حرکت یک خط به بالا در سند
xg, x<, or x Esc+<	رفتن به خط X در سند - برای مثال، تایپ g100 و خط 100am سند را نمایش می‌دهد. اگر X حذف شود، به صورت پیشفرض به خط 1 می‌رود.
xG, x>, or x Esc+>	رفتن به خط X در سند. اگر X حذف شود، به صورت پیشفرض به آخرین خط سند می‌رود.
xp or x%	رفتن به نقطه‌ای X در صد از طریق سند - برای مثال، تایپ p50 به نقطه نیمه سند می‌رود.
/pattern	جستجوی جلو برای الگو در سند، شروع از مکان فعلی. برای مثال، تایپ /BUGS به دنبال رشته BUGS جستجو می‌کند.
?pattern	انجام جستجوی معکوس، پیدا کردن نمونه‌های الگو قبل از مکان فعلی
N or /	تکرار جستجوی قبلی
Q, :Q, or ZZ	خروج از <code>less</code> با <code>q</code>

نکته: ترکیب Esc+V به فشردن کلید Esc به همراه کلید V اشاره دارد.

جدول 5.3 یک بخش کوچک از دستورات موجود در less را نشان می‌دهد. برای یادگیری بیشتر در مورد less، می‌توانید صفحه man آن را مطالعه کنید:

1. به حالت متنی سیستم لاگین کنید یا یک پنجره ترمینال باز کنید.

2. دستور less را تایپ کنید. این کار صفحه man برای less را باز می‌کند.

3. اولین صفحه متن را بخوانید. هنگامی که به آخرین کلمه در پایین صفحه برسید، کلید spacebar را فشار دهید. این کار باعث جایگزینی صفحه به صفحه بعدی می‌شود تا بتوانید مطالعه را ادامه دهید. (می‌توانید به جای spacebar از کلید PageDown یا موارد دیگری که در جدول 5.3 ذکر شده‌اند استفاده کنید. جایگزینی مشابه در مراحل بعدی نیز امکان‌پذیر است).

4. کلید up arrow را فشار دهید. این کار باعث حرکت نمایش به سطر بالا می‌شود که مفید است اگر تنها چند کلمه از انتهای صفحه قبلی را مجددًا بخواهید بخوانید.

5. کلید down arrow را فشار دهید. همانطور که انتظار می‌رود، این کار باعث حرکت نمایش به سطر پایین می‌شود.

6. کلید Esc را فشار داده و به دنبال آن کلید V را بزنید. این کار باعث جایگزینی نمایش به صفحه قبل می‌شود.

7. کلید G را بفشارید تا به انتهای صفحه man بروید.

8. کلید G را (بدون استفاده از کلید Shift) بفشارید تا به ابتدای صفحه man بروید.

9. تایپ کنید /OPTIONS تا به بخش آپشن‌ها بروید. احتمالاً نتایج اولیه شما رفرنس‌های این بخش خواهد بود، تا به خود آپشن‌ها.

10. به صورت تکراری کلید N را فشار دهید تا به بخش گزینه‌ها بروید.

11. کلید Q را بفشارید تا az less و بنابراین از مطالعه صفحه man خارج شوید.

نکته: بعضی از اجرا شدن‌ها حساس به بزرگی و کوچکی حروف جستجو می‌کنند، اما برخی دیگر حساس به این موضوع نیستند. در گام 9، سعی کنید /options را با حروف کوچک جستجو کنید که تعیین شود وضعیت پیاده‌سازی شما چگونه است.

البته، زمانی که صفحات man را می‌خوانید، احتمالاً دقیقاً از این گزینه‌ها استفاده نخواهید کرد؛ شما از هر فیچر مورد نیاز برای یافتن محتوا مورد علاقه خود استفاده خواهید کرد. راه حل کار آشنایی با چند فیچر مهم است تا بتوانید از less به طور موثر برای خواندن صفحات man و مستندات دیگر استفاده کنید.

اگرچه `less` برای خواندن صفحات `man` مهم است، اما می‌توانید از آن برای خواندن سایر اسناد متنی نظیر `README` که با بسیاری از برنامه‌ها همراه هستند یا اسناد متنی ساده دیگر استفاده کنید. برای استفاده از `less` به این شیوه، نام `less` را تایپ کنید و پس از آن نام فایل مورد نظر را برای خواندن وارد کنید، مانند `less README` برای خواندن یک سند `README`. می‌توانید تمامی اقدامات خلاصه شده در جدول 5.3 (یا صفحه `man less`) را بر روی اسناد به این شیوه خوانده شده همانند صفحات `man` استفاده کنید.

## Getting Help Using Info Pages

سیستم صفحات `man` در سیستم عامل‌های مانند Unix، از جمله لینوکس، رایج است، اما این سیستم نسبت به سایر سیستم‌ها که جدیدتر هستند، محدودیت‌هایی دارد. یک سیستم مستندات جدیدتر به نام `info` نیز موجود است. چند صفحه پیش رو توضیح می‌دهند چگونه صفحات `info` در سیستم صفحات `man` کاستی‌ها را پر می‌کنند و اینکه چگونه از صفحات `info` استفاده کنید.

### درک هدف صفحات `info`

طراحی اصلی صفحات `man` به دهه‌ها قبل باز می‌گردد، بنابراین از برخی از توسعه‌های مهم در مدیریت اطلاعات قدیمی تر است. قابل ذکره که صفحات `man` دارای لینک‌های هایپرلینک است. اگرچه بخشی به نام "See Also" در صفحات `man` رایج است، اما نمی‌توانید یکی از این موارد را انتخاب کرده و به صورت مستقیم صفحه `man` مرتبط را بخوانید؛ شما باید از سیستم `man` خارج شده و دستور `man` جدیدی را تایپ کنید تا صفحه جدید را بخوانید. این عدم وجود لینک‌های هایپرلینک است همچنین یک ناهماهنگی در ناوبری درون یک صفحه `man` بزرگ ایجاد می‌کند. شما می‌توانید از جستجوهای متنی برای یافتن اطلاعات مورد نظر استفاده کنید، اما این اغلب منجر به یافتن متن اشتباه می‌شود یا اگر یک رشته را اشتباه تایپ کنید، جستجوی شما ممکن است کاملاً شکست بخورد.

هدف از صفحات `info` این است که این مشکلات را با پشتیبانی از لینک‌های هایپرلینک برطرف کند. هر صفحه `info` به عنوان یک `node` شناخته می‌شود و سیستم صفحات `info` به عنوان یک مجموعه مرتبط از `node`‌ها است، شبیه به شبکه جهانی وب (WWW) اینترنت. مستندات یک برنامه ممکن است در چندین `node` تقسیم شود، که هر `node` ممکن است به راحتی یافته و جستجو شود - اما اگر برای یافتن اطلاعات نیاز به جستجو باشید و نمی‌دانید که اطلاعات در کدام `node` وجود دارد، ممکن است نیاز به جستجو در چندین `node` داشته باشید.

ها در سطوحی سازماندهی شده‌اند که مشابه سطوح سازماندهی در یک کتاب هستند. به عنوان مثال، این کتاب فصول و دو سطح عنوان داخل هر فصل دارد. به همین ترتیب، صفحه `info` برای یک برنامه احتمالاً یک مانند یک فصل خواهد داشت، همراه با چندین `node` در یک سطح پایین‌تر، مشابه عنوان‌های فصل. برخی از صفحات `info` برنامه‌ها شامل سطوح دیگر نیز هستند تا به سازماندهی اطلاعات کمک کنند.

از نظر سبک نگارش، صفحات info مشابه صفحات man هستند - توضیحات مختصر و جامع در مورد موضوعاتشان - برای افرادی که حداقل به طور کلی با برنامه‌های مورد نظر آشنا هستند. اگر شما در سطح ابتدایی شروع با یک برنامه هستید، انواع دیگری از مستندات (که بعداً در "یافتن مستندات اضافی" توضیح داده می‌شوند) ممکن است یک گزینه بهتر باشد.

نکته: در صورت موجودیت در توزیع شما، دایرکتوری‌های /usr/doc و /usr/share/doc اغلب حاوی اطلاعات بسیار مفیدی هستند. اگر نتواستید اطلاعاتی که به دنبال آن هستید را در صفحات man یا info پیدا کنید، در این دایرکتوری‌ها جستجو کنید.

به طور کلی، برنامه‌های حمایت شده توسط بنیاد نرم‌افزار آزاد (FSF) از صفحات info به جای صفحات man استفاده می‌کنند. بسیاری از برنامه‌های FSF اکنون با صفحات man حداقلی همراه هستند که کاربران را به صفحات info برنامه هدایت می‌کنند. برنامه‌نویسان غیر از FSF کمتر به صفحات info توجه کرده‌اند؛ بسیاری از این برنامه‌ها به هیچ وجه با صفحات info همراه نیستند و به جای آن بر روی صفحات man سنتی تکیه می‌کنند. مرورگر info قادر به خواندن و نمایش صفحات man است، بنابراین استفاده انحصاری از info می‌تواند یک استراتژی موثر برای خواندن مستندات استاندارد لینوکس باشد.

## Reading Info Pages

ابزار معمول برای خواندن صفحات info با نام `infonamid` می‌شود. برای استفاده از آن، شما را تایپ کرده و سپس موضوع را وارد می‌کنید، مانند `infonamid info` برای یادگیری در مورد خود سیستم.info. وقتی در سیستم info هستید، می‌توانید از چندین کلید میانبر (خلاصه شده در جدول 5.4) برای حرکت در اطلاعات استفاده کنید.

Table 5.4 info file-navigation commands

Keystroke	Action
?	نمایش اطلاعات راهنمایی
N	حرکت به گره (node) بعدی در یک سری گره‌های مرتبط در یک سطح سلسله مرتبی. این عمل ممکن است لازم باشد اگر نویسنده قصد داشته باشد چندین گره به ترتیب خاصی خوانده شوند.
P	حرکت به عقب در یک سری گره‌های یک سطح سلسله مرتبی. این می‌تواند مفید باشد اگر به جلو حرکت کرده‌اید و نیاز به مرور مطالب قبلی دارید.

U	حرکت به بالا یک سطح در سلسله مراتب گرهها
Arrow keys	حرکت دادن مکان نما در صفحه، امکان انتخاب لینک های گره یا پیمایش صفحه
Page Up, Page Down	این کلیدها به ترتیب بالا و پایین در یک گره واحد پیمایش می کنند. (مرورگر استاندارد info بسیاری از دستورات پیچیده تر که توسط less استفاده می شود و در جدول 5.3 توضیح داده شده را نیز پیاده سازی می کند.)
Enter	حرکت به گره جدید پس از انتخاب آن. لینک ها با ستاره (*) در سمت چپ نام آنها نشان داده می شوند.
L	نمایش آخرین صفحه info که خوانده اید. این عمل می تواند شما را به بالا، پایین یا به سمت های جانبی در سلسله مراتب درخت info ببرد.
T	نمایش صفحه بالای یک موضوع. معمولاً این همان صفحه ای است که از آن وارد سیستم شده اید.
Q	خروج از سیستم صفحه info

به عنوان یک نمونه از استفاده از صفحات info، مراحل زیر را امتحان کنید:

1. به کامپیوتر به حالت متنی لاگین کنید یا یک پنجره ترمینال باز کنید.
2. info را تایپ کنید. باید node اصلی مستندات info ظاهر شود.
3. صفحه اصلی را با استفاده از کلیدهای پایین یا کلید PageDown بخوانید تا صفحه کامل را مشاهده کنید.
4. از طریق کلیدهای پایین، لینک به Expert Info را در نزدیک پایین صفحه اصلی انتخاب کنید.
5. کلید Enter را بزنید تا به Expert Info بروید.
6. کلید U را بزنید تا یک سطح به بالا بروید - به node اصلی.
7. به Advanced node ای که دارد بروید.
8. کلید N را بزنید تا به node بعدی در سطح فعلی، که Expert Info است، بروید.

9. کلید Q را بزنید تا از info خارج شوید.

## Finding Additional Documentation

اگرچه صفحات man و صفحات info هر دو منابع مفیدی هستند، اما مستندات دیگری نیز موجود هستند و گاهی اوقات ترجیح داده می‌شوند. به طور کلی، راهنمایی در لینوکس می‌تواند به سه شکل باشد: مستندات اضافی در کامپیوتر شما، مستندات اضافی آنلاین و کمک از متخصصان.

نکته: دسته‌های مستندات ممکن است با یکدیگر ادغام شوند. به عنوان مثال، ممکن است مستندات آنلاین در دسترس باشند اما تا زمانی که یک بسته مناسب را نصب نکنید، در کامپیوتر شما موجود نباشند.

## Locating Program Documentation on Your Computer

بیشتر برنامه‌های لینوکس با مستندات خود همراه هستند، حتی به طور جدا از صفحات man یا info. در واقع، برخی از برنامه‌ها مستندات زیادی دارند که به عنوان یک بسته جداگانه نصب می‌شود، معمولاً با واژه‌های samba-doc یا doc در نام بسته، مانند documentation یا less

ساده‌ترین و سنتی‌ترین شکل مستندات برنامه، یک فایل به نام README یا README.txt یا چیزهای مشابه است. اینکه دقیقاً چه اطلاعاتی در این فایل وجود دارد، از یک برنامه به برنامه دیگر بسیار متفاوت است. برخی از برنامه‌ها، این فایل به قدری مختصر است که تقریباً بی‌استفاده است. اما برای دیگران، بسیار مفید است. این فایل‌ها تقریباً همیشه فایل‌های متن ساده هستند، بنابراین می‌توانید آنها را با less یا با ویرایشگر متن مورد علاقه‌تان بخوانید.

نکته: فایل‌های README اغلب شامل اطلاعات در مورد ساخت بسته نرم‌افزاری هستند که برای فایل‌های برنامه ارائه شده با یک توزیع معتبر نمی‌آید. اما نگهدارندگان توزیع‌ها به ندرت این اطلاعات را در فایل‌های README خود تغییر می‌دهند.

اگر برنامه را به عنوان یک فایل سورس کد از وبسایت نگهدارنده پکیج نرم‌افزاری دانلود کرده‌اید، فایل README به طور معمول در دایرکتوری build از فایل سورس کد فشرده شده ظاهر می‌شود. اگر برنامه را از یک فایل پکیج نصب کرده‌اید، فایل README ممکن است در یکی از چندین مکان باشد که به توزیع شما بستگی دارد. مکان‌های محتمل عبارتند از:

- usr/doc/packagename/
- usr/share/doc/packagename/
- usr/share/doc/packages/packagename/

در لیست قبلی، `packagename` نام بسته نرم‌افزار است. نام بسته گاهی شامل شماره نسخه است، اما بیشتر موقع شامل نمی‌شود. به علاوه به (یا به جای) فایل `README`، بسیاری از برنامه‌ها فایل‌های مستندات دیگر را ارائه می‌دهند. این ممکن است شامل یک فایل باشد که جزئیات دقیق تاریخچه برنامه را مستند کند، توضیحاتی در مورد روند کامپایل و نصب، اطلاعات در مورد فرمتهای فایل پیکربندی و غیره. با دیدن دایرکتوری ساخت فایل سورس کد یا دایرکتوری که فایل `README` را پیدا کرده‌اید، برای فایل‌های دیگر بررسی کنید.

برخی از برنامه‌های بزرگ با مستندات گسترده در فرمتهای Portable (Document Format) PostScript، Hypertext Markup Language (HTML) یا فرمتهای دیگر ارائه می‌شوند. بسته به فرمت و پکیج، ممکن است یک فایل یا مجموعه بزرگی از فایل‌ها را پیدا کنید. همانند فایل‌های `README`، این فایل‌ها ارزش مشاوره را دارند، به ویژه اگر بخواهید یاد بگیرید چگونه از یک پکیج به بهترین نحو استفاده کنید.

بعضی از برنامه‌ها برای کنترل رفتار خود از فایل‌های پیکربندی استفاده می‌کنند که معمولاً در دایرکتوری `/etc` قرار دارند. اگرچه نحو در فایل‌های پیکربندی اغلب سخت قابل فهم است، اما بسیاری از توزیع‌ها فایل‌های پیکربندی پیش‌فرض را ارائه می‌دهند که شامل توضیحات جزئیات هستند. جزئیات از یک برنامه به برنامه دیگر متفاوت است، اما خطوط توضیحات اغلب با نشانه هشت (#) شروع می‌شوند. شما ممکن است با خواندن توضیحات در فایل پیکربندی یک برنامه، به اندازه کافی اطلاعات جمع‌آوری کنید تا تنظیمات آن را تنظیم کنید.

اگر نتوانید یک فایل `README` یا مشابه آن پیدا کنید، می‌توانید از ابزارهای مختلف برای کمک به یافتن فایل‌های مستندات استفاده کنید. بسیاری از این ابزارهای جستجو، مانند `grep` و `find`، در فصل 8، "earching, Extracting, and Archiving Data" آمده است:

می‌توانید از سیستم مدیریت پکیج توزیع خود (که در جزئیات در فصل 9 توضیح داده شده است) برای یافتن مستندات استفاده کنید. به عنوان مثال، در یک سیستم مبتنی بر RPM، دستور `rpm -ql package | grep find` را تایپ کنید. این دستور برای یافتن مستندات پکیج `package` استفاده می‌شود. استفاده از `grep` برای جستجوی رشته "doc" در فهرست فایل یک ترفند خوب است چرا که دایرکتوری‌های مستندات تقریباً همیشه رشته "doc" را در خود دارند.

دستور `find` لینوکس می‌تواند در سراسر درخت دایرکتوری شما یا یک زیرمجموعه از آن برای یافتن فایل‌هایی که با یک معیار خاص همچوایی دارند، جستجو کند. برای مثال، برای جستجوی یک فایل که شامل یک رشته خاص در نام آن است، می‌توانید دستور `find /usr/share/doc -name "*string"` را وارد کنید که در اینجا "string" کلمه کلیدی است که می‌خواهید در نام یک فایل پیدا کنید. این دستور درخت دایرکتوری `/usr/share/doc/` را جستجو می‌کند، اما می‌توانید به جای این دایرکتوری، درخت دایرکتوری دیگری را جستجو کنید. اگر در یک دایرکتوری با تعداد زیادی فایل و ساب دایرکتوری جستجو کنید، این دستور ممکن است زمان زیادی برای اجرا نیاز داشته باشد.

برنامه whereis برای یافتن فایل‌ها در یک مجموعه محدود از موقعیت‌ها جستجو می‌کند، مانند دایرکتوری‌های استاندارد فایل‌های دودویی، دایرکتوری‌های کتابخانه، و دایرکتوری‌های صفحات راهنمای. این ابزار دایرکتوری‌های کاربر را جستجو نمی‌کند. ابزار whereis یک راه سریع برای یافتن اجرایی‌های برنامه و فایل‌های مرتبط مانند مستندات یا فایل‌های پیکربندی است. برای استفاده از آن، whereis را تایپ کنید و سپس نام دستور یا فایل مورد نظر را وارد کنید، به عنوان مثال whereis less برای یافتن اجرایی less و مستندات مرتبط.

دستور locate لینوکس در یک دیتابیس از نام‌های فایل جستجو می‌کند که توسط لینوکس نگهداری می‌شود. بنابراین، این دستور قادر است وظیفه‌اش را سریعتر از دستور find انجام دهد، اما شما قادر به کنترل بخشی از کامپیوتر که سیستم در آن جستجو می‌کند نیستید. برای استفاده از آن، locate را تایپ کرده و سپس رشته‌ای که می‌خواهید پیدا کنید را وارد کنید، به عنوان مثال locate xterm برای یافتن هر فایل مرتبط با برنامه ترمینال XTerm.

نکته: دیتابیس locate معمولاً هر 24 ساعت یکبار به روزرسانی می‌شود. اگر شما در حال کار با یک توزیع تازه نصب شده هستید یا به دنبال فایل‌های بسته‌ای که به تازگی نصب شده‌اند می‌گردید، دستور updatedb باید با دسترسی مدیر، super user اجرا شود تا پیش از استفاده از دستور locate، دیتابیس locate به صورت دستی به روزرسانی شود.

پس از پیدا کردن فایل‌های مستندات، شما باید بدانید چگونه آنها را بخوانید. جزئیات این موضوع البته به فرمت فایل مستندات بستگی دارد. شما می‌توانید از دستور less برای خواندن بسیاری از فایل‌ها استفاده کنید. بیشتر توزیع‌ها less را به گونه‌ای تنظیم می‌کنند که قادر به تفسیر فرمت‌های معمول فایل، مانند HTML، و همچنین به صورت خودکار فایل‌های ذخیره‌شده به صورت فشرده را باز کند تا فضای دیسک ذخیره شود.

نکته: اگر می‌خواهید یک فایل متنی قالب‌بندی شده مانند یک فایل HTML را به صورت خام ببینید، از گزینه -L- به دستور less استفاده کنید، به عنوان مثال `less -L file.html`.

جدول 5.5 خلاصه‌ای از فرمت‌های متدال فایل مستندات و برنامه‌هایی که می‌توانید برای خواندن آنها استفاده کنید را نشان می‌دهد. استفاده از این فرمت‌ها از یک برنامه به برنامه دیگر متغیر است.

Table 5.5 Common documentation file formats

Filename extension	Description	Programs for reading
.1 through .9	Unix man pages	man, info, less
.gz, .xz, or .bz2	فایلی که با xz، gzip یا 2bzip فشرده شده است.	از دستورات gunzip، unxz یا bunzip 2 برای خارج کردن از حالت فشرده استفاده کنید یا از دستور

		استفاده کنید که ممکن است بتواند فایل را از حالت فشرده خارج کرده و فرمت آن را بخواند.
.txt	Plain text	از دستور less یا هر ادیتور تکست دلخواهی که دارد استفاده کنید.
.html or .htm	HTML	Any web browser
.odt	OpenDocument text	OpenOffice.org, LibreOffice, or many other word processors
.pdf	Portable Document Format (PDF)	Evince, Okular, Adobe Reader, xpdf
.tif, .png, .jpg	Graphics file formats	The GIMP, Eye of GNOME (eog)

نکته: باز کردن یک فایل بصورت دستی با استفاده از دستورات 2bunzip, gunzip, unxz، یا 2bunzip ممکن است نیاز به نوشتن نسخه فشرده نشده آن به دیسک داشته باشد، بنابراین ممکن است نیاز باشد فایل را به دایرکتوری Home خود کپی کنید. برای اطلاعات بیشتر به فصل 8 مراجعه کنید.

## Locating Program Documentation Online

علاوه بر مستنداتی که در کامپیوتر خود پیدا می‌کنید، می‌توانید مستندات را در اینترنت جستجو کنید. اکثر بسته‌ها وبسایت‌های مرتبط با خود دارند که ممکن است در صفحات info، man، فایل‌های README یا مستندات دیگر اشاره شده باشند. بررسی این صفحات برای جستجوی مستندات توصیه می‌شود.

تعدادی از منابع عمومی مستندات لینوکس در اینترنت قابل دسترس هستند. با این حال، معمولاً مستندات ارائه شده توسط این سایتها منقضی است. به طور معمول، توزیع‌ها و وبسایت‌های آنلاین خود را دارند که مستندات به روز را فراهم می‌کنند. منابع مفید مستندات آنلاین عبارتند از:

- پروژه مستندات لینوکس یا LDP در ([tldp.org](http://tldp.org))
- مستندات محصول Red Hat در ([access.redhat.com/documentation](http://access.redhat.com/documentation))
- مستندات رسمی اوبونتو در ([help.ubuntu.com](http://help.ubuntu.com))

- مستندات openSUSE در (doc.opensuse.org)
- مستندات Fedora در (docs.fedoraproject.org)
- مستندات لینوکس Mint در (linuxmint.com/documentation.php)

نکته: توجه داشته باشید که موارد موجود در سایت LDP ممکن است خیلی قدیمی باشند. حتماً تاریخ بازنگری موردنی را قبل از استفاده بررسی کنید!

بسیاری از توزیع‌های لینوکس از افراد داوطلب اجتماعی برای نوشتن و یا بهبود مستندات خود استفاده می‌کنند. تجربه عملی این داوطلبان به دقت و عمق مستندات توزیع افزوده می‌شود، که این وبسایت‌ها را به منابع ارزشمندی تبدیل می‌کند.

## Consulting experts

هر مشکلی که باعث شود شما به دنبال مستندات بگردید، احتمالاً نخستین نفری نیستید که این کار را انجام می‌دهد. در برخی موارد، می‌توانید زمان زیادی را با درخواست کمک از دیگران صرفه‌جویی کنید. برخی منابع خاص عبارتند از:

**توسعه‌دهندگان:** برنامه‌ها بسیاری از نویسندهای اپن سورس خوشحال‌اند که به سوالات پاسخ دهند یا حمایت محدودی فراهم کنند، به ویژه اگر یک باگ باعث مشکل شما شده باشد. پروژه‌های بزرگ (شامل بیشتر توزیع‌های لینوکس) دارای بسیاری از نویسندهای هستند، و این پروژه‌ها اغلب انجمن‌های وب یا لیست‌های پستی ایجاد می‌کنند تا به کاربران و توسعه‌دهندگان کمک کنند تا با یکدیگر ارتباط برقرار کنند.

**انجمن‌های وب و لیست‌های پستی:** این منابع در قالب مختلفی ارائه می‌شوند اما هدف‌های مشابهی دارند: این امکان را فراهم می‌کنند تا کاربران با یکدیگر ارتباط برقرار کرده و تخصص خود را به اشتراک بگذارند. بسیاری از توزیع‌ها انجمن‌های وب اختصاصی دارند؛ برای پیدا کردن انجمن خود، نام توزیع خود را در وب سرچ کنید. لیست‌های پستی برای برنامه‌های فردی معمول‌تر هستند. برای یافتن اطلاعات درباره لیست‌های پستی، به وبسایت اصلی برنامه مراجعه کنید.

**IRC:** یک ابزار برای ارتباط متنی در زمان واقعی بین گروه‌های کوچک افراد است. برای استفاده از IRC، به یک برنامه کاربر IRC نیاز دارید، مانند Irssi، HexChat، Smuxi یا Smuxi. سپس می‌توانید به یک کanal IRC بپیوندید که کاربران IRC در آن به صورت همزمان پیام مبادله می‌کنند.

**مشاوره تخصصی:** پرداخت هزینه به یک مشاور می‌تواند غالباً ارزشمند باشد تا یک مشکل پیچیده را حل کند، به خصوص اگر در مواجهه با یک وضعیت "زمان طلاست" باشید که تأخیر در حل مشکل به معنای واقعی هزینه بیشتری خواهد داشت. جستجو در وب بسیاری از شرکت‌های مشاوره لینوکس را نمایش می‌دهد.

**جستجو در وب:** موتورهای جستجوی وب منابع بسیاری از جمله صفحات `man`، سایت‌های مستندات برنامه، انجمن‌های وب و حتی گفتگوهای کانال IRC را ایندکس می‌کنند. یک جستجو در وب ممکن است شما را به جواب از یک کارشناس برساند بدون نیاز باشد به صورت مستقیم با کارشناس تماس بگیرید.

**نکته:** بعضی از توزیع‌های لینوکس، مانند SUSE Enterprise Linux و Red Hat Enterprise Linux، از پشتیبانی همراه هستند. اگر از چنین توزیعی استفاده می‌کنید، ممکن است قبلًا برای خدمات مشاوره هزینه پرداخت کرده باشید.

استفاده دقیق از این منابع می‌تواند به شما در حل بسیاری از مشکلات لینوکس کمک کند، چه این مشکلات به عدم دانش شما، یک اشتباہ پیکربندی، یک باگ برنامه یا حتی یک فاجعه وحشتناک مثل بروزرسانی نرم‌افزاری باشد که باعث ناتوانی در بوت شدن کامپیوتر شما شده است.

مشکل امروز این است که اغلب اطلاعات زیادی در دسترس است؛ جداسازی اطلاعات غیرمرتبط (یا حتی اطلاعات نادرست) برای یافتن مشاوره مفید ممکن است دشوار باشد. برای پیشگیری از این چالش، بیان موضوع با جزئیات می‌توانید باشد. می‌توانید با افزودن کلمات کلیدی مرتبط به مشکل که کمتر مرسوم هستند، جستجوی وب را به سمت خود جلب کنید. کلمات موجود در پیام‌های خطای می‌توانند به این نظر کمک کنند. اگر یک مشکل را در یک انجمن وب ارسال یا یک گزارش خطای نویسنده یک برنامه ارسال می‌کنید، تا جای ممکن خودتان را با جزئیات شرح دهید. اطلاعاتی در مورد توزیعی که استفاده می‌کنید، نسخه نرم‌افزار و جزئیات خاص در مورد عملکرد برنامه بسیار کمک‌کننده‌اند. اگر برنامه پیام‌های خطای نمایش می‌دهد، آنها را به دقت نقل قول کنید. این جزئیات به متخصصان کمک می‌کنند تا به دلیل مشکل بپردازنند.

# Chapter 6

## Managing Hardware

اگرچه لینوکس یک نرم افزار است، اما برای عملکرد خود به سخت افزار وابسته است. قابلیت ها و محدودیت های سخت افزار شما تأثیرگذار بر قابلیت ها و محدودیت های لینوکس در حال اجرا بر روی آن سخت افزار خواهد بود. بنابراین، شما باید این ویژگی ها را برای هر کامپیوترا که به طور تخصصی از آن استفاده می کنید یا قصد خرید آن را دارید، بدانید.

در این فصل، شما با وظایف مدیریتی اساسی سخت افزار خود آشنا می شوید و آنها را انجام خواهید داد. ما با مسائل سطح پایین شروع می کنیم، مانند ماهیت CPU و مادربرد. منبع تغذیه که اغلب نادیده گرفته می شود، اگر اندازه اش کمتر از حد نیاز باشد یا به درستی عمل نکند مشکلاتی می تواند ایجاد کند، بنابراین ما هم به آن می پردازیم. بعد از آن، به بررسی انواع مختلف هارد دیسک ها که معمولاً هم در ورک استیشن ها و هم سرور پیدا می شوند، می پردازیم و در مورد مراقبت های ویژه مورد نیاز برای راه اندازی آنها صحبت می کنیم. پس از آن، بر روی مانیتورها و درایورهای Display، نقطه مشکل متداول در لینوکس، تمرکز خواهیم کرد. امروزه، بیشتر دستگاه های خارجی از طریق رابط سریال یونیورسال (USB) متصل می شوند، بنابراین این فصل انواع این دستگاه ها را نیز بررسی می کند. در نهایت، این فصل به مسئله متداول درایورها می پردازد، که اجزای نرم افزاری هستند که دستگاه های سخت افزاری را کنترل می کنند.

### Learning About Your CPU

واحد پردازش مرکزی (CPU) شما (گاهی اوقات به آن پردازنده هم گفته می شود) "مغز" کامپیوترا شمامست؛ بیشتر محاسبات واقعی کامپیوترا انجام می دهد. ما در فصل های قبل به موضوع وجود CPU به عنوان توزیع پذیری نرم افزار اشاره کرده ایم. برای اجرای نرم افزار روی یک CPU خاص، بیشتر نرم افزارها باید مجددآ برای آن CPU کامپایل شوند. بنابراین، مهم است که کافی اطلاعاتی در مورد خانواده های مختلف CPU داشته باشید تا نقاط قوت و ضعف آنها را بشناسید و بتوانید نوع CPU مورد استفاده در کامپیوترا خود را تشخیص دهید.

**نکته:** بسیاری از دستگاه ها دارای مدارهای محاسباتی ویژه تر هستند. به ویژه، کارت های گرافیک شامل واحد های پردازش گرافیک (GPU) هستند تا محاسبات گرافیکی ویژه را انجام دهند.

### Understanding CPU Families

تولیدکنندگان CPU معمولاً خطوط محصول خود را با انجام بهبودهای مرتب بر محصولات خود ایجاد می کنند. این بهبودها از جزئی (مانند اجرای CPU با نرخ کلاک بالاتر که مانند اجرای موتور با سرعت بالاتر است) تا بازطرابی های متوسط برای بهبود سرعت (که مانند تغییر قطعات برای به دست آوردن سرعت بهتر است) و بازطرابی های رادیکال (که مانند استفاده از یک موتور بزرگتر یا موتوری کارآمدتر، اما بر اساس طراحی اصلی هستند) متغیر است. تمام این انواع تغییرات در یک خانواده واحد CPU باقی میمانند، بنابراین می توانند

همان کد را مثل مدل های قبلی خود اجرا کنند. تفاوت های بیشتری نیز بین خانواده های CPU وجود دارد؛ دو CPU از خانواده های مختلف به طور معمول نمی توانند برنامه های با اینتری یکدیگر را اجرا کنند، با اینکه استثناءها وجود دارد، یکی از این استثناءها که به زودی توضیح می دهم، بسیار حائز اهمیت است.

در کامپیوترهای دسکتاپ، دو خانواده CPU متداول هستند:

**x86**: این نوع CPU از مدل 8086 شرکت اینتل نشأت گرفت، اما اولین مدل قابل اجرای لینوکس مدل 80386 (همچنین به نام 386 شناخته می شود) بود. توسعه این CPU با مدل 80486 (همچنین به نام 486)، پنتیوم و AMD CPU های مرتبط اینتل مانند سلرون ادامه یافت. شرکت های دیگر مانند VIA، Cyrix، AMD و دیگران همگی CPU هایی را منتشر کرده اند که با طراحی های اینتل سازگار هستند.

**x86-64**: نام های دیگر برای این معماری شامل x، 64AMD64 و EM64T معماري  $64\times 86$  را به عنوان یک گسترش 64 بیتی به معماری  $86\times$  ایجاد کرد. به طور فوق العاده ای، CPU های  $64\times 86$  قادر به اجرای کد 32 بیتی  $86\times$  اولیه هستند، اما هنگام اجرا در حالت 64 بیتی، این CPU ها به امکانات اضافی دسترسی دارند که سرعت را بهبود می بخشنند. اینتل نیز CPU های  $64\times 86$  خود را ایجاد کرده است. هم اینتل و هم AMD از نام های خط محصول یکسان برای CPU های  $64\times 86$  خود استفاده کرده اند که برای CPU های  $86\times$  هم استفاده شده اند. این واقعیت می تواند سخت باشد تا بتوانید تشخیص دهید حداقل بر اساس نام تجاری، آیا یک  $86CPU$  که 32 بیتی است دارید یا یک  $64\times 86CPU$  64 بیتی دارید.

برای اکثر کامپیوترهای دسکتاپ و لپتاپ ها، شما به طور معمول با یکی از پردازنده های خط Intel Core مواجه می شوید:

i3: ارزان ترین پردازنده، سرعت و عملکرد پردازشی پایین را ارائه می دهد.

i5: پردازنده میانه رده، سرعت و عملکرد پردازشی متوسط را به همراه قیمت متوسط ارائه می دهد.

i7: پردازنده مرتبه بالا، سرعت و عملکرد پردازشی پیشرفته را ارائه می دهد اما با قیمت بالاتر.

### CPU Bit Depth

واحدهای پردازش مرکزی (CPU) داده ها را به صورت با اینتری (با مبنای 2) پردازش می کنند، به این معنی که اعداد فقط با استفاده از دو رقم 0 و 1 نمایش داده می شوند. با این حال، CPU ها به حداقل اندازه اعدادی که می توانند پردازش کنند محدودیت دارند، و این محدودیت ها با استفاده از تعداد رقم های با اینتری یا بیت CPU می توانند پردازش کند، توصیف می شوند. به عنوان مثال، یک مدل 32 بیتی می تواند اعدادی را پردازش کند که شامل حداقل 32 رقم با اینتری هستند. این به عنوان اعداد صحیح مثبت بیان می شود، به این معنا که اعداد می توانند از 0 تا  $32^2 - 1$  (یا 4,294,967,295) در پایه 10 که افراد به طور عمومی از آن استفاده می کنند) متغیر باشند. هنگام مواجه با اعداد بزرگتر، CPU باید دو یا چندین عدد را ترکیب کند، که این امر نیازمند کد اضافی است.

CPU های با اعمق بیت بزرگتر در کار با RAM های زیاد مزیت دارند، زیرا آدرس های RAM باید در اندازه واحد پایه اصلی CPU جا بیافتدند. مخصوصاً، یک CPU مدل 32 بیتی محدودیت 4 گیگ بیت (GiB) حافظه دارد – با این حال برخی معما رها، از جمله x86، ترفندهایی برای دور زدن برخی محدودیت ها فراهم می کنند. افزایش عمق بیت سرعت را به جز در کار با اعداد بسیار بزرگ بهبود نمی بخشد؛ با این حال، معما ری CPU مدل 64 بیتی x86-64 به دلیل دلایل غیر مرتبط با پیشینه 32 بیتی x86 خود سریعتر است.

علاوه بر CPU های x86 و x64، چندین خط مدل دیگر نیز موجود است. بیشتر این CPU ها در کاربردهای تعبیه سازی یا در سرورهای بسیار پیشرفته استفاده می شوند. در زیر چند مدل که ممکن است بر آنها برخورد کنید آورده شده اند:

**ARM**: پردازنده Advanced RISC Machine یا ARM در سال 1985 معرفی شد، اما اخیراً به عنوان پردازنده اصلی در دستگاه های کوچک مانند تبلت ها و سیستم های کامپیوتر تعبیه شده به کار گرفته شده است. مخفف RISC در اسم به Reduced Instruction Set Computing اشاره دارد، که به معنای پشتیبانی از تعداد کمتری دستور العمل است. این به این معناست که کمترین تعداد ترانزیستورها برای سیستم لازم است، که در نتیجه به کاهش مصرف انرژی توسط تراشه CPU منجر می شود. این ویژگی است که پردازنده های ARM را برای دستگاه های کوچک که از لینوکس استفاده می کنند، ایده آل می کند.

**PowerPC**: این پردازنده به عنوان یک تلاش همکارانه از سوی اپل، آی‌بی‌ام و موتورولا ایجاد شد و از سال 1994 تا 2006 پردازنده اصلی اپل بوده و در کامپیوترهای مک این دوره استفاده شده است. امروزه، در برخی از کنسول های بازی، دستگاه های تعبیه شده و سرورها استفاده می شود. هر دو نسخه 32 و 64 بیتی از این معما ری موجود است. PPC یک اختصار متداول برای این معما ری است.

**Itanium**: اینتل معما ری Itanium یا IA-64 را به عنوان یک جایگزین 64 بیتی برای خط x86 ایجاد کرد؛ با این حال، جز در زمینه سرور، در بازار کمترین نفوذ را به خود اختصاص داد.

لینوکس قابلیت اجرا بر روی پردازنده های PowerPC، Itanium و ARM را دارد، همچنین بر روی خانواده های Scalable Microprocessor without Interlocked Pipelined Stages (MIPS) یا SPARC Processor Architecture (SPARC) نیز قابل اجرا است. با این حال، انتخاب شما احتمالاً در انتخاب توزیع محدود خواهد شد. زیرا نرم افزار باید برای معما ری های جدید مجدداً کامپایل و تست شود، آماده سازی یک توزیع برای هر معما ری نیاز به زحمت دارد و بسیاری از مدیران توزیع ممکن است تمایل یا امکانات لازم را برای بیش از x86 و x64 صرف نکنند.

نکته: توزیع لینوکس Debian بر روی بسیاری از معما ری ها در دسترس است، بنابراین اگر نیاز به پشتیبانی از لینوکس بر روی یک دسته وسیع از پردازنده ها دارید، استفاده از Debian ممکن است منطقی باشد.

بسیاری از پردازنده ها امروزه مدل های چند هسته ای هستند. این پردازنده ها مدارهای مربوط به دو یا چند پردازنده را در یک واحد بسته بندی می کنند. زمانی که در یک مادربرد قرار می گیرند، یک چنین پردازنده ای برای سیستم عامل به عنوان چندین پردازنده به نظر می آید. مزیت این است که لینوکس می تواند هر تعداد برنامه

صرف کننده CPU که دارد، اجرا کند و این برنامه‌ها با رقابت برای منابع CPU، سرعت یکدیگر را به میزان قابل توجهی کاهش نخواهند داد.

نکته: برخی از مادربردهای سطح بالا نیز از چندین پردازنده پشتیبانی می‌کنند، بنابراین می‌توانید برای مثال، دو پردازنده 4 هسته‌ای را استفاده کنید تا به عنوان یک سیستم با پردازنده 8 هسته‌ای عملکرد داشته باشد.

## Identifying Your CPU

اگر یک سیستم لینوکس کارکرده دارید، می‌توانید با استفاده از سه دستور متنی در کامند لاین شل، اطلاعات زیادی درباره پردازنده‌تان بدست آورید:

با تایپ این دستور، اطلاعات اولیه درباره هسته و پردازنده نمایش داده می‌شود. به عنوان مثال،  
**uname -a**: یکی از سیستم‌های من اطلاعاتی مانند "AMD Phenom(tm) II X3 700e Processor" را  
برمی‌گرداند که در میان دیگر اطلاعات، تولید کننده و شماره مدل پردازنده را نشان می‌دهد.

**lscpu**: دستور lscpu اطلاعات اضافی در حدود 20 خط خروجی را بازمی‌گرداند. بسیاری از این اطلاعات فنی سطح بالاست، مانند اندازه‌های حافظه Cache که پردازنده پشتیبانی می‌کند. بخشی از این اطلاعات کمتر فنی است، مانند معماری و تعداد پردازنده‌ها یا هسته‌هایی که پشتیبانی می‌شود.

**cat /proc/cpuinfo**: دستور cat /proc/cpuinfo اطلاعات بیشتری نسبت به lscpu بازمی‌گرداند. احتمالاً شما به این اطلاعات خود نیاز نخواهید داشت، اما یک توسعه‌دهنده یا تکنسین ممکن است بخواهد برخی از این اطلاعات را برای رفع مشکلات استفاده کند.

یک نکته مهم این است که پردازنده‌های مدرن x86-64 قادر به اجرای نرم‌افزارهای کامپایل شده برای معماری قدیمی‌تر x86 هستند. بنابراین، ممکن است یک توزیع لینوکس 32 بیتی را بر روی یک پردازنده 64 بیتی اجرا کنید. خروجی این دستورها در چنین مواردی ممکن است گیج‌کننده باشد. به عنوان مثال، اینجا بخشی از آنچه lscpu در یکی از چنین سیستمی نشان می‌دهد آمده است:

Architecture: i686

CPU op-mode(s): 64-bit

نیاز نشان می‌دهد که پردازنده از نوع 86x 686 است (یک نوع خاص از این معماری است)، اما CPU op-mode (s) نیاز نشان می‌دهد که پردازنده از عملیات 64 بیتی پشتیبانی می‌کند. اگر مشکلی در تفسیر این خروجی داشته باشد، ممکن است با جستجوی مدل پردازنده در وبسایت تولید کننده، به اطلاعاتی بررسی کنید. با این حال، تولید کنندگان عمدتاً این اطلاعات را در برگه‌های مشخصات دشوار به خواندن مخفی می‌کنند، بنابراین آماده باشید با دقت بخوانید.

## Identifying Motherboard Capabilities

اگر پردازنده مغز کامپیووتر باشد، مادربرد باقی سیستم عصبی مرکزی کامپیووتر است. مادربرد یک برد مدار بزرگ داخل کامپیووتر است که توسط یک چیپست کنترل می شود، که یک یا چند چیپ است که وظایفهای کلیدی برای کامپیووتر ارائه می دهند؛ آنها به پردازنده در ارتباط با رابطهای هارد دیسک، رابطهای USB، دستگاههای شبکه و غیره کمک می کنند. برخی از چیپستها دارای مدار ویدئویی برای کارت های گرافیک هستند، اگرچه این قابلیت گاهی اوقات جداگانه است و گاهی اوقات در پردازنده جای گرفته است.

نکته: گاهی اوقات به مادربردها به عنوان "mainboards" نیز اشاره می شود.

علاوه بر چیپست، مادربردها شامل رابطهای قابل اتصال برای اجزاء کلیدی هستند:

- یک یا چند اسلات برای پردازندههای کامپیووتر
- اسلات های حافظه RAM
- اسلات های کارت های PCI یا دیگر کارت ها
- کانکتورهای هارد دیسک های داخلی مانند رابط SATA
- کانکتورهای پنل پشتی که رابطهای خارجی برای دستگاههای USB، صفحه کلید، نمایشگرها و غیره فراهم می کنند
- کانکتورها برای دستگاههای خارجی اضافی مانند پلاگین های USB پنل جلویی؛ این دستگاهها از طریق کابل های داخلی کوتاه متصل می شوند.

برخی از مادربردها که معمولاً در کامپیووترهای دسکتاپ بزرگ و سرورها استفاده می شوند، دارای کانکتورهای زیادی برای مقاصد مختلف هستند. این گونه مادربردها قابلیت ارتقاء بالایی دارند، اما اندازه آنها به حدی بزرگ است که نیاز به کیس های بزرگ دارند. مادربردهای دیگر کوچکتر هستند و می توانند در کامپیووترهای کوچک استفاده شوند، اما چنین کامپیووترهایی امکانات ارتقاء کمتری دارند. همچنین لپ تاپ ها نیز دارای مادربرد هستند که ضرورتاً از نوع کوچک هستند و امکانات ارتقاء داخلی کمتری دارند.

بیشتر کانکتورهای موجود در یک مادربرد توسط چیپست اصلی آن مدیریت می شوند. برخی از مادربردهای سطح بالا کانکتورهایی برای ویژگی های خارج از قابلیت های چیپست اصلی ارائه می دهند. این ویژگی ها نیاز به چیپست ثانویه دارند، مانند یک چیپست اترنت اضافی برای یک پورت شبکه دوم یا یک چیپست اضافی SATA برای رابطهای هارد دیسک بیشتر یا سریع تر.

برای فهمیدن این که ویژگی توسط چیپست اصلی یا چیپست ثانویه فراهم شده باشد، می توانید با استفاده از دستور `lspci` در مورد اکثر ویژگی های مادربرد مطلع شوید که اطلاعات در مورد دستگاههای PCI را نشان می دهد. خروجی آن چیزی شبیه به مثال مختصر زیر خواهد بود:

```
$ lspci
```

```
00:00.0 Host bridge: Advanced Micro Devices [AMD] RS780 Host Bridge
```

00:11.0 SATA controller: ATI Technologies Inc SB700/SB800 SATAm  
Controller [IDE mode]

00:12.0 USB Controller: ATI Technologies Inc SB700/SB800 USB OHCI0m  
Controller

00:14.1 IDE interface: ATI Technologies Inc SB700/SB800 IDE  
Controller

00:14.2 Audio device: ATI Technologies Inc SBx00 Azalia (Intel HDA)

01:05.0 VGA compatible controller: ATI Technologies Inc Radeon HD  
3200m  
Graphics

01:05.1 Audio device: ATI Technologies Inc RS780 Azalia controller

02:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd.m  
RTL8111/8168B PCI Express Gigabit Ethernet controller (rev 02)

03:06.0 Ethernet controller: Intel Corporation 82559 InBusiness  
10/100m  
(rev 08)

ممکن است شما همه چیز در این خروجی را به طور کامل درک نکنید، اما باید بتوانید برخی اطلاعات را از آن استخراج کنید. به عنوان مثال، کامپیوتر دارای چندین دستگاه ATI، یک کنترلکننده SATA، یک کنترلکننده USB، یک کارت گرافیک وغیره است. دو دستگاه اینترنت وجود دارد، یکی توسط Realtek ساخته شده و دیگری توسط Intel. با اینکه از این خروجی این موضوع واضح نیست، اما آداپتور شبکه Realtek به مادربرد داخلی است و در حالتی که دستگاه Intel روی یک کارت پلاگین قرار دارد.

## Sizing Your Power Supply

منبع تغذیه یا پاور کامپیوتر جریان متناوب (AC) را از پریز دیواری می‌گیرد و آن را به جریان مستقیم (DC) تبدیل می‌کند که مادربرد شما و هر چیزی که به آن متصل می‌کنید از آن استفاده می‌کند. لپتاپ‌ها و برخی از واحدهای کوچک دسکتاپ از "Bricks" یا آداپتور برق استفاده می‌کنند که می‌توانید آن را در زمین قرار دهید. کامپیوترهای دسکتاپ بزرگ دارای واحدهای داخلی منبع تغذیه هستند. این واحدهای داخلی بزرگتر هستند، هم از نظر اندازه فیزیکی و هم از نظر مقدار توانی که می‌توانند ارائه دهند.

هر منبع تغذیه محدودیت‌هایی در مقدار توان جریان مستقیم (DC) که می‌تواند ارائه دهد دارد. این امر اهمیت دارد زیرا هر دستگاهی که به کامپیوتر وصل می‌شود، مقدار خاصی از توان مصرف می‌کند. اگر تولید کننده کامپیوتر اقدام به صرفه‌جویی کند، منبع تغذیه ممکن است برای کامپیوتر هنگام تحویل به سختی کافی باشد. اگر یک هارد دیسک یا یک کارت پلاگین با مصرف بالا اضافه کنید، ممکن است مقدار توانی که منبع تغذیه می‌تواند ارائه دهد، را فراتر ببرید. نتیجه ممکن است عملکرد ناپایدار باشد و کامپیوتر ممکن است خراب شود یا به طور ناقص عمل کند، حتی شاید داده‌ها یا فایل‌ها را خراب کند. اینگونه مشکلات ممکن است از مشکلات دیگر مانند حافظه (RAM) ناسالم یا یک هارد دیسک ناتوان سخت تر شناسایی شوند.

اگر نیاز به تعویض منبع تغذیه خود دارید، به خروجی آن به وات توجه کنید. شما باید قادر باشید خروجی منبع تغذیه فعلی خود را در یک استیکر پیدا کنید اما برای بیشتر سیستم‌های دسکتاپ، حداقل باید کامپیوتر خود را باز کنید. اطمینان حاصل کنید که منبع تغذیه جدید حداقل برای تعداد وات‌های منبع تغذیه فعلی شما ارزیابی شده است. همچنان، اطمینان حاصل کنید که جا می‌شود. ابعاد به طور استاندارد تعیین شده‌اند، اما چندین نوع متغیر وجود دارد. در صورت استفاده از لپتاپ یا کامپیوتر دسکتاپ کوچک با منبع تغذیه خارجی، باید اطمینان حاصل کنید که منبع تغذیه جایگزین نوع صحیحی از کانکتور را به کامپیوتر فراهم کند. در این موارد، بهترین راه حل معمولاً خرید منبع تغذیه جایگزین از تولید کننده کامپیوتر است.

## Understanding Disk Issues

دیسک‌ها بخش اساسی از بیشتر نصب‌های لینوکس هستند، بنابراین اهمیت دارد که چگونه عمل می‌کنند و چگونه لینوکس با آن‌ها تعامل دارد. این بخش سه مسئله اساسی مرتبط با دیسک را توضیح می‌دهد: رابطه‌ای سخت‌افزاری دیسک، پارتیشن‌بندی دیسک و فایل‌سیستم‌ها. همچنین، برخی از مسائل مرتبط با دیسک‌های پرتاپل، از جمله دیسک‌های نوری (Blu-ray و CD-ROM و DVD-ROM) را توضیح می‌دهد.

نکته: شما می‌توانید لینوکس را در یک پیکربندی بدون دیسک نصب کنید، که در آن یک کامپیوتر لینوکس با استفاده از فایل‌های ذخیره شده در یک سرور شبکه بوت می‌شود.

## Disk Interfaces

امروزه، سه رابط دیسک متداول هستند:

**PATA**: این رابط در گذشته بسیار متداول بوده است، اما در حال فراموش شدن است. این رابط دارای کابل‌های 40 یا 80 پین با عرض وسیع است که همزمان چندین بیت داده را انتقال می‌دهند؛ به همین دلیل که کلمه موازی در نام آن (Parallel ATA) وجود دارد. یک کابل PATA می‌تواند حداکثر سه کانکتور داشته باشد؛ یکی برای مادربرد یا کارت کنترلر دیسک و دو کانکتور دیگر برای حداکثر دو هارد دیسک. نام‌های جایگزین برای PATA (یا نسخه‌های خاص از آن) شامل IDE یا EIDE یا ATAPI می‌باشد. استاندارد ATAPI یک رابط نرم‌افزاری تعریف می‌کند که به ATA در مدیریت دستگاه‌هایی به جز هارد دیسک کمک می‌کند. اگرچه در برخی موارد تفاوت‌های مهم بین تکنولوژی‌های توصیف شده توسط این اصطلاحات نسخه‌های متغیر ممکن است مهم باشد، اما امروزه این اصطلاحات معمولاً به صورت مترادف استفاده می‌شوند.

**SATA**: در سال 2003، نسخه‌ای سریال از پروتکل ATA ایجاد شد، بنابراین Serial ATA یا SATA تقریباً سازگار با PATA از نظر نرم‌افزار است، اما از کابل‌های نازکتری استفاده می‌کند که تنها یک هارد دیسک را بر روی هر کابل می‌تواند هندل کند. در سال 2012، SATA به فناوری دیسک مهم در کامپیوترهای جدید تبدیل شد. نسخه خارجی آن، یعنی eSATA، ارتباطات با سرعت بالا به هارد دیسک‌های خارجی فراهم می‌کند.

**NVMe**: رابط Non-volatile Memory Express یا NVMe خصوصاً برای پشتیبانی از استاندارد ذخیره‌سازی درایوهای SSD طراحی شده است. دستگاه‌های SSD از پرتوها و سرهای مغناطیسی برای خواندن یا نوشتن داده‌ها استفاده نمی‌کنند؛ به جای این‌که داده‌ها به صورت الکترونیکی مانند حافظه ذخیره شوند، از چیپ‌های حافظه فلاش استفاده می‌کنند که پس از خاموش شدن برق از سیستم، داده‌ها را حفظ می‌کنند. به دلیل این‌که داده‌ها به صورت الکترونیکی خوانده و نوشته می‌شوند و نه به صورت مغناطیسی، سرعت دسترسی به طور قابل توجهی سریع‌تر است و درایوهای SSD مانند هارد دیسک‌های PATA و SATA آسیب‌پذیر نیستند. در ابتدا، درایوهای SSD کوچک و گران‌قیمت بودند، اما اخیراً بزرگ‌تر و ارزان‌تر شده‌اند، که آنها را به عنوان یک جایگزین واقعی برای هارد دیسک‌های فیزیکی در بسیاری از محیط‌ها می‌سازد.

علاوه بر این فناوری‌ها، دیگر فناوری‌ها نیز وجود دارند. SCSI یک رابط موازی است که یک زمانی بر روی سرورها و رابطه‌ای مرتبط با دستگاه‌های پیشرفته بسیار معمول بود، اما امروزه کمتر معمول است. رابط موازی متصل به سریال (SAS) یک نسخه سریال است که خیلی شبیه به SATA است. هر دو این تکنولوژی مهم هستند زیرا ATAPI بر اساس SCSI مدل شده است. USB معمولاً برای اتصال دیسک‌های خارجی استفاده می‌شود.

نکته: بیشتر توزیع‌های لینوکس مدرن، دیسک‌های SAS، SATA و USB را از نظر نرم‌افزاری به عنوان دیسک‌های SCSI می‌شناسند و یک فایل دستگاه در دایرکتوری /dev/ ایجاد می‌کنند که با `x` sd شروع می‌شود، که `x` حرف درایو است و از `a` شروع می‌شود. با این حال، برای سیستم‌هایی که از رابط NVMe جدید استفاده می‌کنند، لینوکس این فایل‌های دستگاه را به عنوان `x` nvme ایجاد می‌کند.

## Partitioning a Disk

می‌توانید یک هارد دیسک را به عنوان یک مجموعه از سکتورها فرض کنید، هر کدام از آنها مقدار کوچکی از داده‌ها را نگه می‌دارند؛ به طور معمول 512 بایت، با این‌که برخی از دیسک‌ها سکتورهای بزرگ‌تری دارند. سخت‌افزار هارد دیسک به ندرت به خودش کمک می‌کند تا داده‌ها را بر روی دیسک سازماندهی کند، به جز این‌که وسیله‌ای برای خواندن و نوشتن سکتورهای خاص فراهم می‌کند. مدیریت داده‌ها بر روی دیسک خود به سیستم عامل سپرده می‌شود. پارتیشن‌های دیسک و فایل‌سیستم‌ها دو سطح سازماندهی معمول بر روی دیسک‌ها هستند که به مدیریت داده‌هایی که در آنها ذخیره شده‌اند، کمک می‌کنند.

پارتیشن‌ها بسیار شبیه به دراورهای یک کابینت فایل هستند. یک دیسک تنها را به عنوان کابینت اصلی فایل در نظر بگیرید که به چندین پارتیشن تقسیم شده است، مانند دراورها. این تشبيه تا حدودی خوب است، اما محدودیت‌های خود را دارد. برخلاف دراورهای کابینت فایل، پارتیشن‌های دیسک می‌توانند با اندازه و تعداد دلخواه، در حد اندازه دیسک، ایجاد شوند. یک دیسک معمولاً بین یک تا دوازده پارتیشن دارد، با این حال می‌توانید بیشتر بسازید.

پارتیشن‌های دیسک برای کمک به تقسیم دیسک به قطعاتی با اهداف گستردگر و متفاوت به وجود آمدند. مانند پارتیشن‌های مختلف برای سیستم‌عامل‌های مختلف یا برای انواع مختلف داده‌ها در داخل یک سیستم‌عامل. به عنوان مثال، معمول است پارتیشن‌های جداگانه برای فضای Swap (که در صورت تمام شدن حافظه RAM مانند حافظه RAM استفاده می‌شود)، برای فایل‌های داده کاربران و برای خود سیستم‌عامل ایجاد شود.

هارد دیسک‌ها و پارتیشن‌های آن‌ها اغلب در نمودارهای مشابه شکل 6.1 نمایش داده می‌شوند. این نمودار پارتیشن‌ها را به عنوان زیربخش‌های دیسک نشان می‌دهد، با اندازه نسبی حجم پارتیشن‌ها نسبت به حجم حقیقی آنها بر روی دیسک. بنابراین، در شکل 6.1 می‌توانید ببینید که /root نسبت به فضای Swap حافظه بزرگتر است. همانطور که در شکل نیز واضح است، پارتیشن‌ها بخش‌های پیوسته‌ای از یک دیسک هستند، به عبارت دیگر یک پارتیشن نمی‌تواند در داخل یک پارتیشن دیگر باشد.

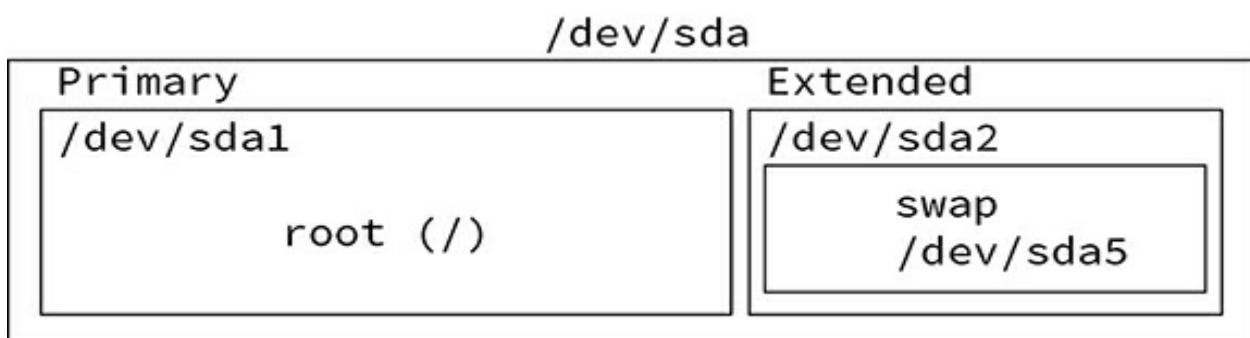


Figure 6.1 Disk partitions are often visualized as boxes within a hard disk.

**نکته:** بعضی از ابزارهای پارتیشن‌بندی پارتیشن‌های خود را به صورت یک دسته عمودی نشان می‌دهند به جای زنجیره افقی. اصل هر دو روش یکسان است.

raigترین برنامه پارتیشن‌بندی برای کامپیوترهای 86x و 64-86x در طول سالها با اسمی مختلفی نظیر مستر بوت ریکورد (MBR)، ام‌اس-داس (MS-DOS) و پارتیشن‌بندی BIOS شناخته شده است و سه نوع پارتیشن را پشتیبانی می‌کند:

**Primary:** این نوع پارتیشن ساده‌ترین نوع پارتیشن است. یک دیسک می‌تواند از صفر تا چهار پارتیشن اصلی داشته باشد، که یکی از آنها می‌تواند یک پارتیشن Extended باشد.

**Extended:** این نوع ویژه‌ای از پارتیشن‌های اصلی است که به عنوان یک نگهدارنده برای پارتیشن‌های منطقی عمل می‌کند. یک دیسک حداقل یک پارتیشن گستردگی می‌تواند داشته باشد.

**Logical:** این پارتیشن‌ها داخل یک پارتیشن گستردگی قرار دارند. در تئوری، یک دیسک می‌تواند میلیاردها پارتیشن منطقی داشته باشد، که بدین ترتیب محدودیت چهار پارتیشن اصلی را برطرف می‌کند، اما در عمل احتمالاً بیش از حدود دوازده تا آنها را نخواهید دید.

استفاده از سه نوع پارتیشن توسط MBR ناشیانه و محدودکننده است، اما مقاومت به تغییر آن را در مدت سه دهه حفظ کرده است. اما یک محدودیت سخت در پارتیشن‌های MBR وجود دارد: آن‌ها نمی‌توانند از دیسک‌های بزرگتر از 2 تی‌بايت (تبی‌بايت) پشتیبانی کنند، با فرض سکتورهای 512 بايتی که تقریباً امروزه جهانی است.

جدول پارتیشن (GPT) یک جانشین برای MBR است. GPT از دیسک‌های تا 8 زدی‌بايت (زیبی‌بايت) پشتیبانی می‌کند، که تقریباً 4 میلیارد برابر حد MBR است. همچنین، GPT حداقل 128 پارتیشن را به طور پیش‌فرض پشتیبانی می‌کند، بدون تفاوت بین پارتیشن‌های اصلی، گستردۀ و منطقی. در این جوانب، GPT یک سیستم پارتیشن‌بندی برتر نسبت به MBR است؛ با این حال، پشتیبانی آن بین سیستم‌عامل‌ها متفاوت است. لینوکس هر دو سیستم را به خوبی پشتیبانی می‌کند. ویندوز زمانی که کامپیوتر از سیستم ورود/خروج پایه (BIOS) استفاده می‌کند از MBR فقط برای بوت شدن استفاده می‌کند، و فقط زمانی از GPT برای بوت شدن استفاده می‌کند که کامپیوتر بر اساس UEFI است. بنابراین، اگر با ویندوز Dual-Boot باشد، ممکن است نیاز به دقت در انتخاب سیستم پارتیشن‌بندی داشته باشد.

## Multibyte Units

استفاده از پیشوندهای واحدهای سیستم بین‌المللی واحدهای SI - کیلو (k)، مگا (M)، گیگا (G)، ترا (T) و غیره - به اشتراک با بايت (B) برای ارجاع به مقادیر بزرگ فضای ذخیره‌سازی معمول است، مانند kB، MB، kB و غیره. به لحاظ فنی، این واحدها به عنوان مقادیر مبنای 10 تعریف شده‌اند؛ کیلو به معنای 1,000، مگا به معنای 1,000,000 و غیره. با این حال، در کامپیوترها، مقادیر مبنای 2 به عنوان مثال  $2^{10}$  (1024) و  $2^{20}$  (1,048,576) غالباً طبیعی‌تر هستند، بنابراین واحدهای SI اغلب (اما همیشه نه) برای نمایش این مقادیر مبنای 2 استفاده شده‌اند. این عمل موجب ایجاد ابهام شده است، زیرا همیشه واضح نیست که آیا از واحدهای مبنای 10 یا مبنای 2 استفاده می‌شود.

برای حل این مشکل، انجمن مهندسان برق و الکترونیک (IEEE) یک مجموعه جدید از پیشوندها به عنوان IEEE-1541 تعریف کرد. در این سیستم، واحدها و پیشوندهای جدید مقادیر مبنای 2 را توصیف می‌کنند. چند نمونه از این واحدها به شرح زیر هستند:

- یک کیبی بايت (KiB) برابر با  $2^{10}$  (1024) بايت است.
- یک مبی بايت (MiB) برابر با  $2^{20}$  (1,048,576) بايت است.
- یک گیبی بايت (GiB) برابر با  $2^{30}$  (1,073,741,824) بايت است.
- یک تبی بايت (TiB) برابر با  $2^{40}$  (1,099,511,627,776) بايت است.

در این کتاب، هنگام توضیح ویژگی‌هایی که بهتر است با استفاده از این سیستم بیان شوند، ما از واحدهای IEEE-1541 استفاده می‌کنیم، مانند محدودیت‌های اندازه جدول پارتیشن. بیشتر ابزارهای دیسک لینوکس از واحدهای SI و IEEE-1541 به درستی استفاده می‌کنند، اما اینکه کدام یک استفاده می‌شود، به

دلخواه نویسندها بستگی دارد. به این تفاوت توجه کنید، به ویژه هنگام مواجهه با اعداد بزرگ؛ توجه داشته باشید که یک تبی بایت تقریباً 10 درصد بزرگتر از یک ترابایت است.

چندین سیستم پارتیشن‌بندی دیگر وجود دارد، اما احتمالاً با اکثر آن‌ها روبرو نخواهید شد. یک استثناء احتمالی، نقشه پارتیشن‌اپل (APM) است که اپل آن را در کامپیوترهای مک خود قبل از تغییر به پردازنده‌های اینتل استفاده می‌کرد.

زمانی که به پارتیشن‌بندی یک دیسک میرسیم، لینوکس از سه خانواده ابزار پشتیبانی می‌کند:

خانواده **fdisk**: ابزارهای `fdisk`، `cfdisk` و `sfdisk` ابزارهای متň ساده‌ای برای پارتیشن‌بندی دیسک‌های MBR و برخی از انواع جدول پارتیشن خارجی‌تر هستند. این ابزارها به خوبی کار می‌کنند و امکان بازیابی از برخی خطاهای دیسک را فراهم می‌کنند، اما ماهیت متň آن‌ها ممکن است برای کسانی که با پارتیشن‌بندی دیسک آشنا نیستند، ترسناک باشد

ابزارهای مبتنی بر **libparted**: این ابزارها از انواع جداول پارتیشن از جمله GPT و چندین نوع دیگر پشتیبانی می‌کنند. برخی از این ابزارها، مانند GNU Parted، به صورت متň هستند، اما باقی مانند GParted رابط گرافیکی (GUI) دارند و بنابراین برای کاربران تازه کار احتمالاً آسان‌تر استفاده می‌شوند. تصویر 6.2 نشان‌دهنده فعالیت GParted است. توجه داشته باشید که نمایش آن به ساختار نشان داده شده در تصویر 6.1 شبیه است. بسیاری از نصاب‌های لینوکس ابزارهای پارتیشن‌بندی مبتنی بر `libparted` را که در زمان نصب سیستم اجرا می‌شوند، شامل می‌شوند.

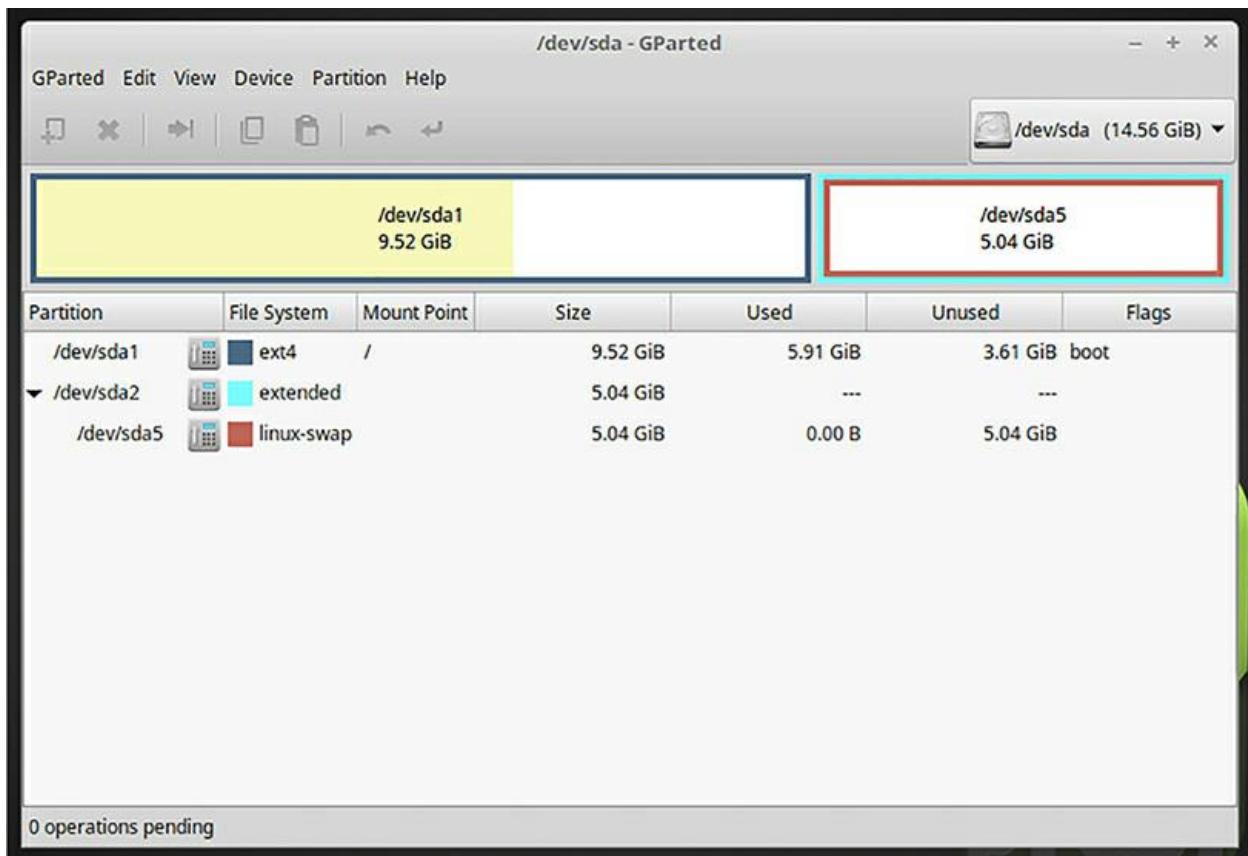


Figure 6.2 GParted, like other GUI disk partitioning tools, provides a graphical representation of your partitions.

خانواده ابزارهای **GPT fdisk**: شامل ابزارهای cgdisk و gdisk است که بر اساس خانواده fdisk مدل‌سازی شده‌اند اما با دیسک‌های GPT کار می‌کنند. این ابزارها گزینه‌های بیشتری برای مدیریت GPT ارائه می‌دهند اما به قیمت آسانی برای کاربران تازه کار.

نکته: در گذشته، ابزار fdisk بر روی دیسک‌های MBR متمرکز بود و ابزار gdisk بر روی دیسک‌های GPT. اما اخیراً توسعه‌دهندگان برای هر دو ابزار با پشتیبانی از هر دو نوع دیسک آزمایش‌ها انجام داده‌اند. توزیع لینوکس شما ممکن است شامل یک نسخه از یک ابزار باشد که هر دو نوع دیسک MBR و GPT را پشتیبانی می‌کند.

اگر با یک سیستم لینوکس از پیش نصب شده کار می‌کنید، ممکن است نیازی به پارتیشن‌بندی دیسک نداشته باشید؛ اما اگر هارد دیسک را تعویض یا نصب مجدد کنید، باید قبل از استفاده از آن، آن را پارتیشن‌بندی کنید. همچنین ممکن است نیاز به پارتیشن‌بندی دیسک‌های پرتابل داشته باشید، اگرچه این‌ها به طور کلی از کارخانه با یک پارتیشن بزرگ پیش‌پردازشده می‌آیند.

برای پارتیشن‌بندی یک دیسک، باید نام فایل دستگاه دیسک را بدانید. در لینوکس، این نام‌های فایل معمولاً /dev/sda و غیره هستند، با هر دیسک یک حرف جدید می‌گیرند. شماره‌گذاری پارتیشن‌ها با

شماره 1 شروع می‌شود، بنابراین ممکن است به `/dev/sdb` و غیره ارجاع دهید. هنگام استفاده از MBR، پارتیشن‌های 1 تا 4 برای پارتیشن‌های اصلی یا گستردۀ اختصاص داده شده‌اند، در حالی که پارتیشن‌های منطقی با شماره‌های 5 به بالا آمده‌اند.

## Understanding Filesystem Issues

بیشتر پارتیشن‌های دیسک حاوی سیستم‌های فایل هستند که ساختارهای داده‌ای هستند که به کامپیووتر کمک می‌کنند تا دایرکتوری‌ها و فایل‌های شما را سازماندهی کند. در ویندوز، هر سیستم‌فایل نقطه دستگاه خود را دریافت می‌کند. در روزهای قدیم که دیسک‌های فلاپی به طور معمول استفاده می‌شدند، حروف درایو A و B برای درایوهای دیسک فلاپی اختصاص داده شده بودند، و این استاندارد همچنان ادامه دارد. حرف درایو C برای پارتیشن اول هارد دیسک (معمولًاً پارتیشن بوت) استفاده می‌شود، حرف D برای دیسک‌های USB خارجی و غیره. با این تفاوت که در لینوکس، تمام سیستم‌های فایل جزو یک درخت دایرکتوری واحد هستند. سیستم‌فایل اصلی به عنوان سیستم‌فایل روت (/) شناخته می‌شود. اگر یک دیسک چندین پارتیشن سیستم‌فایل داشته باشد، هرکدام در یک نقطه‌ی مانند در سیستم‌فایل روت (/) مانند می‌شوند؛ به عبارت دیگر، محتويات سیستم‌فایل‌های اضافی در دایرکتوری‌های خاصی همچون /home (که فایل‌های داده کاربران را نگه داری می‌کند) یا /boot (که فایل‌های بوت را نگه داری می‌کند) در دسترس قرار می‌گیرند. چندین سیستم‌فایل لینوکس وجود دارد، هرکدام با ویژگی‌های منحصر به فرد خود:

نکته: گاهی اوقات کلمه سیستم‌فایل به ساختار دایرکتوری به طور کلی نیز اطلاق می‌شود، حتی اگر شامل چندین سیستم‌فایل سطح پایین باشد. به طور معمول، معنی مورد نظر از زمینه جاری و متن به وضوح مشخص می‌شود.

**ext2fs**: یا دومین سیستم فایل گستردۀ، در دهه 1990 محبوب بود، اما امروزه به دلیل عدم وجود یک ژورنال، که یک ویژگی فایل سیستم است که بررسی‌های فایل سیستم را پس از قطع برق یا خطاهای سیستم سریع‌تر می‌کند، به ندرت استفاده می‌شود. اگرچه، ژورنال فضای دیسک را مصرف می‌کند؛ پس ext2fs هم بر روی دیسک‌های کوچک مفید است. به عنوان مثال، ممکن است بخواهید آن را برای یک بخش جداگانه / استفاده کنید، زیرا چنین بخش‌هایی نسبتاً حجم کمی دارند. کد نوع فایل سیستم لینوکس آن ext2 است.

نکته: extfs در اوایل دهه 1990 استفاده می‌شد، اما به سرعت توسط ext2fs جایگزین شد. دیگر پشتیبانی نمی‌شود.

**ext3fs**: این سیستم فایل در واقع ext2fs با یک ژورنال است. تا حدود سال 2010، این سیستم فایل بسیار محبوب بود، اما ext4fs جایگزین آن شده است. این سیستم فایل از فایل‌های تا حداقل 2 ترابایت و فایل‌سیستم‌های تا حداقل 16 ترابایت پشتیبانی می‌کند (ext2fs محدودیت‌های مشابه را اعمال می‌کند). کد تایپ لینوکس آن ext3 است.

**Ext4fs**: این فایل سیستم یک توسعه سیستم فایل ext است. این سیستم فایل افزایش سرعت و توانایی کنترل فایل‌ها و دیسک‌های بزرگ‌تر را افزوده است؛ فایل‌ها ممکن است تا 16 ترابایت و فایل‌سیستم‌ها ممکن است تا 1 ایگابایت (۲<sup>۶۰</sup> بایت) باشند. ابزارهای لینوکس به آن ext 4 اشاره می‌کنند.

**ReiserFS**: این فایل سیستم، با ابزارهای لینوکس به نام ext3fs، مشابه با ext3fs در ویژگی‌هاست، با محدوده اندازه فایل 8 ترابایت و محدوده اندازه فایل‌سیستم 16 ترابایت. بهترین ویژگی آن، توانایی استفاده بهینه از فضای دیسک با فایل‌های کوچک است؛ فایل‌هایی با اندازه‌های کمی باشد پایین. توسعه ReiserFS کمی کاهش یافته است، اما همچنان قابل استفاده است.

**JFS**: این فایل سیستم را IBM برای سیستم‌عامل AIX خود توسعه داد و در نهایت کد آن به لینوکس منتقل شد. JFS حداکثر اندازه فایل و فایل‌سیستم را به ترتیب 4 پتاپایت و 32 پتاپایت (1 پتاپایت برابر با 1024 ترابایت) پشتیبانی می‌کند. JFS محبوبیت مشابه سایر سیستم‌های فایل لینوکس ندارد. ابزارهای لینوکس از jfs به عنوان کد تایپ آن استفاده می‌کنند.

**XFS**: شرکت سیلیکون گرافیکس این سیستم فایل را برای سیستم‌عامل IRIX خود توسعه داد و بعدها کد آن را به لینوکس اهدا کرد. XFS حداکثر اندازه فایل و فایل‌سیستم را به ترتیب 8 اگزابایت و 16 اگزابایت پشتیبانی می‌کند، که آن را به انتخابی مناسب برای آرایه‌های دیسک بسیار بزرگ می‌کند. XFS با فایل‌های چندرسانه‌ای و فایل‌های پشتیبانی بزرگ خوب کار می‌کند. کد تایپ آن xfs است.

**Btrfs**: فایل‌سیستم جدید (تلفظ: "باتر-اف-اس" یا "بتر-اف-اس") به عنوان فایل‌سیستم لینوکس نسل بعدی طراحی شده است. این فایل‌سیستم از فایل‌های تا 16 اگزابایت و فایل‌سیستم‌های با همان اندازه پشتیبانی می‌کند. همچنین این امکان را فراهم می‌کند که چندین دیسک فیزیکی را به یک فایل‌سیستم ترکیب کنید. تا به این لحظه، Btrfs هنوز در حال آزمایش است، اما ممکن است هنگام کامل شدن بهترین میزان ویژگی‌ها را ارائه دهد. کد تایپ لینوکس آن btrfs است.

اگر لینوکس جدیدی دارید، باید فایل‌سیستم‌های Btrfs، ext4 یا XFS را در نظر بگیرید. در حال حاضر، ext4fs بهترین ویژگی‌ها و عملکرد کلی را ارائه می‌دهد، و Btrfs و XFS نیز برای حجم‌هایی که به ترتیب فایل‌های خیلی کوچک و خیلی بزرگ را در بر دارند، قابل در نظر گرفتن هستند. با این حال، ext4fs یک انتخاب خوب برای حجم‌هایی است که فایل‌های بزرگ دارند، بنابراین می‌توانید از ext4fs برای همه چیز استفاده کنید و به مشکلات زیادی بر نخورید، به ویژه در یک کامپیوتر چند منظوره.

**نکته:** امکان استفاده از چندین فایل‌سیستم در یک نصب لینوکس وجود دارد تا از مزایای هر فایل‌سیستم برای مجموعه‌های مختلفی از فایل‌ها استفاده شود.

علاوه بر فایل‌سیستم‌های اصلی لینوکس، این سیستم‌عامل از چندین فایل‌سیستم دیگر نیز پشتیبانی می‌کند، برخی از آن‌ها در شرایط خاص مهم هستند:

**FAT**: این فایل‌سیستم استاندارد DOS و ویندوز تا ویندوز Me بود. تقریباً همه سیستم‌عامل‌ها از آن پشتیبانی می‌کنند. سازگاری آن این فایل‌سیستم را به گزینه مناسبی برای تبادل داده بین دو سیستم‌عامل

روی یک کامپیوتر تبدیل کرده است. برخلاف بیشتر فایل سیستم‌ها، FAT دو کد تایپ لینوکسی دارد: msdos و vfat. استفاده از msdos باعث می‌شود لینوکس از فایل سیستم به همان شکلی که DOS انجام داده است، با نام‌های کوتاه حداکثر 8 حرف و یک پسوند 3 حرفی (نام‌های فایل 8.3) استفاده کند؛ وقتی از vfat استفاده می‌شود، لینوکس از نام‌های طولانی در FAT پشتیبانی می‌کند.

نکته: سادگی و پشتیبانی گسترده از FAT، این فایل سیستم را به یک فایل سیستم محبوب بر روی فلاش‌درایوهای USB، تلفن‌های همراه، کتابخوان‌های مجازی، مدیای دوربین دیجیتال و غیره تبدیل کرده است.

**NTFS:** مایکروسافت فایل سیستم جدید فناوری (NTFS) را برای ویندوز NT توسعه داد و این فایل سیستم به فایل سیستم پیش‌فرض نسخه‌های جدید ویندوز تبدیل کرد. در اوایل، لینوکس تنها یک درایور خواندن محدود برای NTFS فراهم کرده بود، اما اکنون یک درایور کامل خواندن/نوشتن با نرم‌افزار ntfs-3g در دسترس است. احتمالاً این را بر روی یک بخش بوت ویندوز در یک پیکربندی Dual Boot یا در هارد دیسک‌های پرتابل یا اکسترنال بزرگ خواهید یافت. در لینوکس، درایور کرنل قدیمی به نام ntfs شناخته می‌شود، در حالی که درایور ntfs-3g به نام ntfs-3g نامیده می‌شود.

نکته: درایور ntfs لینوکس در کرنل قرار دارد که آن را سریع می‌کند. در مقابل، درایور ntfs-3g، برخلاف بیشتر درایورهای فایل سیستم، بر پایه کرنل نمی‌باشد، بنابراین سرعت آن کمتر است و نیاز به قدرت پردازش اضافی دارد.

**HFS:** اپل از فایل سیستم سلسله‌مراتبی خود (HFS) در Mac OS تا نسخه 9.x استفاده می‌کرد و هنوز هم در macOS آن را پشتیبانی می‌کند. ممکن است با HFS در برخی از مدیاهای پرتابل و به خصوص در دیسک‌های قدیمی که تحت نسخه‌های قبل از X از Mac OS ایجاد شده‌اند، روبرو شوید. لینوکس از پشتیبانی کامل از HFS برای خواندن/نوشتن با استفاده از درایور hfs خود ارائه می‌دهد.

**+HFS:** یا همان Mac OS Extended، فایل سیستم فعلی برای macOS است؛ ممکن است با آن در کامپیوترهای Mac دوال بوت و برخی از مدیاهای پرتابلی که برای استفاده با کامپیوترهای Mac ایجاد شده‌اند، روبرو شوید. لینوکس با استفاده از درایور hfsplus، پشتیبانی خواندن/نوشتن از +HFS را فراهم می‌کند؛ با این حال، پشتیبانی از نوشتمن به صورت پیش‌فرض در نسخه‌های فایل سیستمی که شامل یک ژورنال می‌شوند، غیرفعال است.

نکته: کاربران مک اغلب از فایل سیستم FAT بر روی رسانه‌های پرتابل استفاده می‌کنند به دلیل دلایل سازگاری.

**ISO-9660:** فایل سیستم ISO-9660 بر روی مدیاهای نوری، به ویژه CD-ROM و CD-R استفاده می‌شود. این فایل سیستم در چندین سطح با قابلیت‌های مختلف وجود دارد. دو افزونه به نام‌های Joliet و Rock Ridge، پشتیبانی از نام‌های فایل بلند را با استفاده از استانداردهای ویندوز و یونیکس به ترتیب فراهم می‌کنند. لینوکس تمام این نوع‌ها را پشتیبانی می‌کند. برای مانت کردن یک دیسک ISO-9660، می‌توان از کد تایپ 9660iso استفاده کرد.

**UDF**: فرمت دیسک یونیورسال (UDF) یک فایل سیستم است که قرار است جایگزین ISO 9660 شود. این فرمت بیشترین بار بر روی مدیاهای DVD و بلوری (Blu-ray) یافت می‌شود، اگرچه گاهی اوقات بر روی CD-R نیز استفاده می‌شود. کد تایپ لینوکس برای این فرمت به طبیعت udf است.

بیشتر فایل سیستم‌های غیر لینوکس، پشتیبانی از مالکیت و دسترسی‌های نوع یونیکسی که لینوکس استفاده می‌کند را ندارند. بنابراین، ممکن است نیاز باشد از گزینه‌های خاص مانند برای تنظیم مالکیت و Rock Ridge دسترسی‌ها استفاده کنید. استثنایها شامل ISO+HFS و ISO 9660 هنگام استفاده از افزونه‌های Rock Ridge است. دیسک‌های Rock Ridge به طور کلی با مالکیت و دسترسی‌هایی که امکان استفاده عادی از دیسک را فراهم می‌کنند، ایجاد می‌شوند. اما اگر با یک دیسک HFS+ روبرو شوید، ممکن است ببینید که ارزش‌های یوزر ID یا (UID) با ارزش‌های کاربران لینوکس شما مطابقت ندارند. بنابراین، ممکن است نیاز باشد داده‌ها را به عنوان root کپی کنید، یک حساب با ارزش UID مطابق ایجاد کنید، یا مالکیت فایل‌ها را در دیسک macOS تغییر دهید. (این گزینه آخر احتمالاً ناپسند خواهد بود اگر قصد دارید دیسک را دوباره تحت استفاده کنید).

برای دسترسی به یک فایل سیستم، از دستور mount استفاده کنید. به عنوان مثال، برای ماندن فایل سیستم روی /5dev/sda در /shared، دستور زیر را وارد کنید:

```
# mount /dev/sda5 /shared
```

در ضمن، می‌توانید یک ورودی برای فایل سیستم در فایل /etc/fstab ایجاد کنید که اطلاعاتی مانند فایل دستگاه، نوع فایل سیستم و نقطه مانند را ذخیره می‌کند. وقتی از یک فایل سیستم استفاده کردید، می‌توانید آن را با دستور umount آنمانند کنید، مانند `umount /shared`.

نکته: دستور umount نامش تنها یک حرف n دارد.

## Using Removable and Optical Disks

اگر یک دیسک پرتابل را به یک کامپیوتر که از اکثر توزیع‌های لینوکس مدرن استفاده می‌کند، وارد کنید، احتمالاً سیستم آن را تشخیص می‌دهد، دیسک را در یک زیرشاخه از /media/ مانند کرده و یک فایل منیجر بر روی دیسک راهاندازی می‌کند. این رفتار باعث می‌شود که سیستم به شیوه‌ای کار کند که برای کاربران ویندوز یا macOS آشنا است.

وقتی از دیسک استفاده کردید، باید آن را آنمانند کنید و سپس با احتیاط جدا کنید. بیشتر فایل منیجرها به شما این امکان را می‌دهند که این کار را با راست‌کلیک بر روی ورودی دیسک در پنل سمت چپ انجام دهید و گزینه‌ای به نام Eject یا Safely Remove یا Unmount کنید، همانطور که در شکل 6.3 نشان داده شده است. اگر این کار را انجام ندهید، فایل سیستم ممکن است آسیب ببیند.

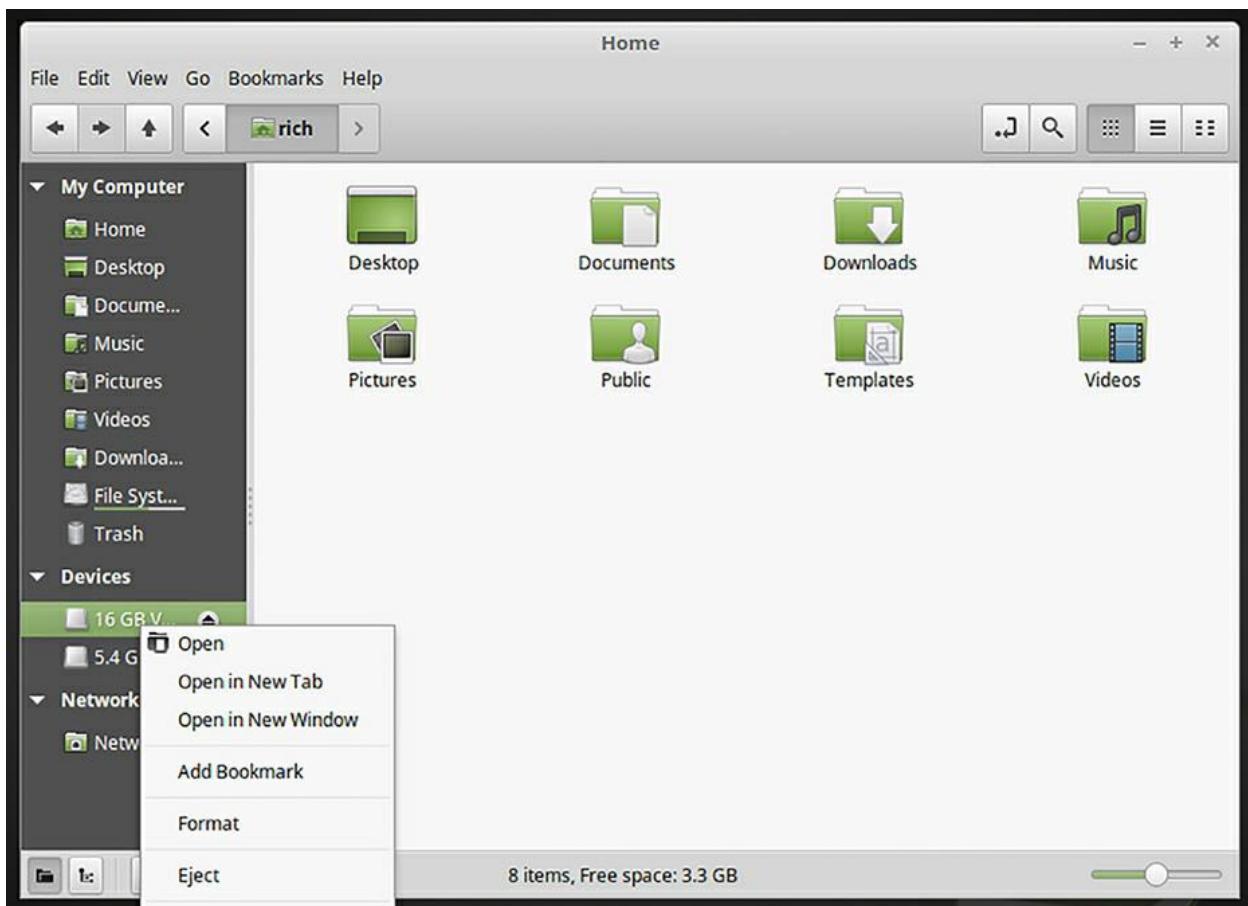


Figure 6.3 Linux file managers enable you to unmount removable media.

نکته: برخی از دستگاه‌ها، مانند دیسک درایوهای نوری (DVD یا CD)، می‌توانند مکانیسم خروج خود را قفل کنند تا از جدا شدن اجباری مدیاهای جلوگیری کنند.

بیشتر دیسک‌های پرتابل یا بدون پارتیشن هستند یا یک پارتیشن دارند. آنها غالباً از سیستم فایل FAT استفاده می‌کنند که یک انتخاب مناسب برای سازگاری با چند پلتفرم است. اگر نیاز دارید، می‌توانید درایوهای فلش USB و بیشتر مدیاهای قابل حمل دیگر را بخشندی کنید.

دیسک‌های نوری در واقعیت نیاز به فایل سیستم‌های خاص خود (UDF یا ISO-9660) دارند. با اینکه می‌توانید این دیسک‌ها را همانند دیسک‌های دیگر مانت و آنماحت کنید، اما فقط می‌توانید اطلاعات را از آن‌ها بخوانید و نمی‌توانید بنویسید. اگر می‌خواهید یک دیسک نوری را بر روی مدیای خالی ایجاد کنید، باید از نرم‌افزارهای خاص مانند record، cdrecord، growisofs، mkisofs یا X-CD-Roast در حالت گرافیکی استفاده کنید. این ابزارها یک فایل سیستم ISO-9660 از فایل‌های مشخص شده ایجاد می‌کنند و سپس آن را بر روی دیسک خالی ذخیره می‌کنند. پس از آن، می‌توانید دیسک را به روش معمول مانت کنید.

## Managing Displays

لینوکس دو حالت Display فراهم می‌کند: حالت متنی و گرافیکی. نمایش متنی به نسبت ساده‌تر است و نیاز به مدیریت کم یا هیچی ندارد. نمایش‌های گرافیکی پیچیده‌تر هستند. در لینوکس، سیستم پنجره X (یا X به اختصار) نمایش گرافیکی را مدیریت می‌کند. این بخش توضیح می‌دهد X چیست و چگونه با سخت‌افزارهای نمایش متداول ارتباط برقرار می‌کند.

## Understanding the Role of X

بیشتر افراد به نرم‌افزار بک گراند اسکرین کامپیوترهای خود فکر زیادی نمی‌کنند؛ همه چیز به نظر می‌رسد که عالی کار می‌کند. البته، پشت پرده، وظایف مدیریت نمایش یک فرایند نسبتاً پیچیده است. برخی از وظایفی که نرم‌افزار باید انجام دهد، در هر پلتفرمی عبارتند از:

- مرحله‌ای که کارت گرافیک را راه اندازی اولیه می‌کند، که شامل تنظیم رزولوشن آن می‌شود.
- تخصیص بخش‌هایی از نمایشگر برای نگهداری پنجره‌های تعلق گرفته به برنامه‌ها.
- مدیریت پنجره‌هایی که به یکدیگر همپوشانی دارند تا فقط محتوای پنجره "بالاتر" نمایش داده شود.
- مدیریت یک نشانگر که توسط کاربر از طریق ماوس یا دستگاه مشابه کنترل می‌شود.
- هدایت ورودی کاربر از یک صفحه کلید به برنامه‌ای که فعال است.
- نمایش متن و اشکال ساده درون پنجره‌ها به درخواست برنامه‌های کاربر.
- فراهم کردن عناصر رابط کاربری برای جابه‌جایی و تغییر اندازه پنجره‌ها.
- مدیریت داخل پنجره‌ها مانند نمایش منوها و نوارهای اسکرول.

در لینوکس، وظایف اول تا شش توسط X (سیستم پنجره‌ای X) انجام می‌شوند، اما وظیفه هفتم توسط یک برنامه به نام Window Manager و وظیفه هشتم توسط کتابخانه‌های گرافیکی معروف به Widget Sets انجام می‌شود. عنصر نمایش فونت در وظیفه ششم ممکن است توسط X انجام شود، اما در سال‌های اخیر بسیاری از برنامه‌ها از یک کتابخانه به نام Xft برای این وظیفه استفاده کرده‌اند. بنابراین، کلیه وظایف مربوط به نمایش در چندین برنامه تقسیم شده است، هرچند X بیشتر وظایف سطح پایین را انجام می‌دهد.

نکته: محیط‌های دسکتاپ شامل Window Manager می‌شوند، اما ویندوز نمی‌جرها بدون محیط دسکتاپ نیز موجود هستند.

توزیع‌های لینوکس مدرن از یکی از دو پکیج نرم‌افزاری X محبوب استفاده می‌کنند:

- X.org
- Wayland

هر دو پکیج نرم‌افزاری نسخه‌ای از X را فراهم می‌کند که به صورت خودکار توانایی شناسایی سخت‌افزار شما را دارد، از جمله کارت گرافیک، مانیتور، صفحه کلید و موس، و خود را به صورت خودکار پیکربندی می‌کند. نتیجه این است که نرم‌افزار به طور معمول بدون نیاز به پیکربندی صریح به درستی کار می‌کند. با این حال، گاهی اوقات این پیکربندی خودکار شکست می‌خورد. وقتی این اتفاق می‌افتد، شما باید به صورت دستی فایل‌های پیکربندی X را ویرایش کنید. برای پکیج X.org، به فایل /etc/X11/xorg.conf، برای Wayland،

تنظیمات پیکربندی برای هر حساب کاربری به صورت جداگانه در فایل config/weston.ini/. ذخیره می‌شود. اگر فایل پیکربندی X.org وجود نداشته باشد، می‌توانید یک فایل نمونه را با اجرای دستور زیر (هنگامی که X در حال اجرا نیست) به عنوان روت ایجاد کنید:

```
# Xorg -configure
```

نکته: فصل 10، "Editing Files" درباره ویرایشگرهای متنه nano و Vi در حالت متنه صحبت می‌کند.

نتیجه این دستور به طور معمول یک فایل به نام /root/xorg.conf.new است. شما می‌توانید این فایل را به /etc/X11/xorg.conf کپی کنید و شروع به ویرایش آن بر اساس نیازهای خود کنید. این کار پیچیده است و خارج از دامنه این کتاب است، اما دانستن نام فایل می‌تواند به شما کمک کند تا شروع کنید؛ شما می‌توانید فایل را مطالعه کرده و با جستجو بر روی آن نام، اطلاعات تکمیلی را پیدا کنید.

## Using Common Display Hardware

بخش زیادی از چالش در ارتباط با دستگاه‌های گرافیک در مدیریت درایورهای مربوط به چیپست‌های گرافیک است. اکثر کامپیوترهای مدرن از یکی از معده‌دود درایورهای گرافیک استفاده می‌کنند:

- درایورهای nvidia و nv با سخت‌افزار گرافیک NVIDIA کار می‌کنند.
- درایورهای fglrx و ati با سخت‌افزار گرافیک AMD/ATI کار می‌کنند.
- درایور intel با سخت‌افزار گرافیک Intel کار می‌کند.
- درایورهای vesa و fbdev درایورهای جامع هستند که با انواع گسترهای از سخت‌افزارها کار می‌کنند، اما بهره‌وری غیربهینه‌ای را تولید می‌کنند.
- بسیاری از کارت‌های گرافیک قدیمی از درایورهای کمتر شناخته‌شده‌تری استفاده می‌کنند.

درایورهای nvidia و fglrx درایورهای انحصاری تولیدکنندگان خود هستند. جزئیات را از وبسایت‌های تولیدکنندگان بررسی کنید یا برنامه‌های نصب این درایورها را جستجو کنید. این درایورهای انحصاری ویژگی‌هایی را ارائه می‌دهند که در همتای اپن سورس خود موجود نیستند، با این حال درایور nouveau برخی از این ویژگی‌ها را پیاده‌سازی می‌کند. در این زمینه، "ویژگی‌های" درایور گرافیک به بهبود عملکرد تفسیر می‌شود، به ویژه از نظر گرافیک D3 و نمایش‌های Real-Time (مانند پخش فایل‌های ویدئویی).

در گذشته، اکثر کارت‌های گرافیک از طریق یک کابل 15 پینی VGA به مانیتور متصل می‌شدند. امروزه، کابل‌های 19 پینی HDMI بسیار متداول هستند. HDMI از مزیت یک واسطه دیجیتال برخوردار است که می‌تواند یک نمایش تمیزتر را در مانیتورهای مدرن با فناوری LED ایجاد کند.

رزولوشن مانیتور به طور معمول با تعداد افقی و عمودی پیکسل‌ها اندازه‌گیری می‌شود. در گذشته، رزولوشن‌هایی با ابعاد به نسبت کم مانند 640×480 رایج بودند، اما امروزه کمتر مشاهده می‌شود که از یک مانیتور با رزولوشن بهینه کمتر از 1024x1280 768x1366 و رزولوشن‌های 1080x1920 یا بالاتر که رایج

هستند استفاده شود. برای تعیین رزولوشن بهینه مانیتور خود، به دفترچه راهنمای مانیتور مراجعه کنید. به طور کلی، مانیتورهای با ابعاد بزرگتر رزولوشن بالاتری دارند؛ با این حال، این موضوع همیشه صدق نمی کند.

نکته: امروزه، برای تطبیق نمایش فیلم‌ها، بیشتر مانیتورها از نسبت ابعاد 16:9 استفاده می‌کنند که به نسبت طول به ارتفاع نمایشگر اشاره دارد. مانیتورهای قدیمی از نسبت ابعاد 4:3 استفاده می‌کردند که استاندارد تلویزیون قبل از ورود محصولات کیفیت بالا بود.

در بهترین حالت ممکن، لینوکس به طور خودکار نسبت به شناسایی دقیق‌ترین رزولوشن مانیتور شما اقدام کرده و هر بار که سیستم را راهاندازی می‌کنید، خود را به این مقدار تنظیم خواهد کرد. متأسفانه در برخی موارد این عملکرد ممکن است کار نکند. اکستنشن کیبورد/ویدیو/موس (KVM) و کابل‌های اکستنشن گاهی ممکن است با این شناسایی خودکار مداخله کنند، و مانیتورهای قدیمی ممکن است از ارتباط لازم بین کامپیوتر و مانیتور پشتیبانی نکنند. ممکن است نیاز به مطالعه دستورالعمل مانیتور باشد تا بهترین رزولوشن آن را بدانید. در بخش ویژگی‌ها یا مشخصات این اطلاعات را جستجو کنید؛ معمولاً به آن optimum resolution یا موارد مشابه گفته می‌شود. همچنین ممکن است شامل مقدار نرخ فرکانسی (Refresh) شود، مانند 1024x1280 و 60 هرتز.

در اکثر موارد، شما می‌توانید رزولوشن را با استفاده از ابزار گرافیکی مانند آیتم Displays در پنل تنظیمات سیستم GNOME (که در شکل 6.4 نشان داده شده است) تنظیم کنید. در شکل 6.4، گزینه Resolution امکان تنظیم رزولوشن به هر مقدار نظر را فراهم می‌کند. اگر نتوانید رزولوشن بهینه را در لیست پایین‌آورده پیدا کنید، ممکن است نیاز به انجام تنظیمات پیشرفته‌تری با ویرایش فایل /etc/X11/xorg.conf داشته باشید، که یک موضوع خارج از دامنه این کتاب است. در موقع نادری، ممکن است نیاز به ارتقاء کارت گرافیک خود داشته باشید؛ برخی از کارت‌ها قادر به پشتیبانی از رزولوشن بهینه مانیتورهای برخی نیستند.

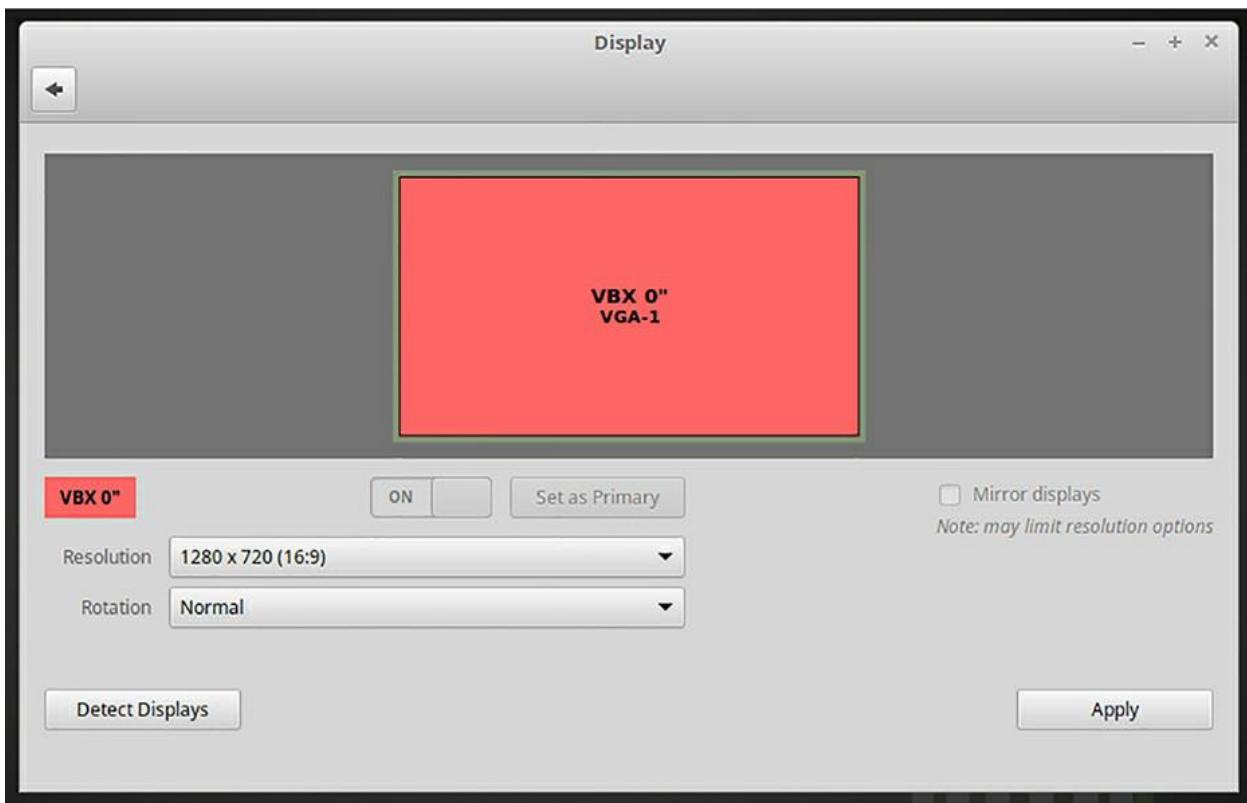


Figure 6.4 Most desktop environments provide GUI tools to help you set your display's resolution.

## Handling USB Devices

بیشتر کامپیوترهای مدرن از عنوان رابط اصلی برای دستگاههای جانبی خارجی استفاده می‌کنند. کیبوردها، ماوس‌ها، دوربین‌ها، حافظه‌های فلش، هارد دیسک‌ها، آداتورهای شبکه، اسکنرها، پرینترها و موارد دیگر همگی می‌توانند از طریق پورت USB متصل شوند. بیشتر اوقات، دستگاههای USB به صورت Plug-and-Play عمل می‌کنند - شما آن‌ها را وصل می‌کنید و آن‌ها کار می‌کنند. برخی از نکات خاص عبارتند از:

نرم‌افزار X معمولاً هر نوع صفحه کلید، ماوس، تبلت و دستگاه مشابهی را هنگامی که آن‌ها را وصل می‌کنید مدیریت می‌کند. اگر مشکلی دارید، ممکن است نیاز به تنظیمات بیشتر در پیکربندی X داشته باشید، اما این یک موضوع پیشرفت‌هه است که خارج از دامنه این کتاب است.

ما دستگاههای فلش USB، هارد دیسک‌های خارجی و سایر دستگاههای ذخیره‌سازی را در این دسته قرار می‌دهیم. همانطور که قبلًا توضیح داده شد در "Using Removable and Optical Disks"، این حیاتی است که پیش از جدا کردن کابل USB آن را Unmount کنید. عدم انجام این کار ممکن است منجر به تخریب داده (Data Corruption) شود.

**Cell Phones, Cameras, e-Book Readers, and Music Players**: شما می‌توانید از اغلب این دستگاه‌ها به عنوان دستگاه‌های دیسک برای انتقال عکس‌ها، موزیک یا سایر فایل‌ها استفاده کنید. با این حال، ممکن است نیاز باشد یک گزینه را در دستگاه فعال کرده تا آن را به شکل دستگاه دیسک برای کامپیوتر نشان دهد. وقتی انجام شد، دستگاه را Unmount کنید و حالت رابط آن را غیرفعال کنید.

**Scanners**: لینوکس از نرم‌افزار SANE به عنوان نرم‌افزار اصلی برای مدیریت اکثر اسکنرها استفاده می‌کند. با این حال، برخی از اسکنرها ممکن است به نرم‌افزارهای ناشناخته یا متعلق به شرکت‌های خصوصی نیاز داشته باشند.

**Printers**: اکثر توزیع‌های لینوکس هنگامی که یک پرینتر USB را وصل می‌کنید به طور خودکار پرینتر مناسب را پیکربندی می‌کنند. بیشتر توزیع‌های لینوکس از سیستم چاپ Unix متداول (CUPS) برای شناسایی و پیکربندی پرینترها استفاده می‌کنند. اگر نیاز به تنظیمات جزئی دارید، تلاش کنید که آدرس 631:<http://localhost> را در مرورگر وب روی کامپیوتر مربوطه وارد کرده تا به رابط کاربری اصلی CUPS دسترسی پیدا کنید. این کار یک ابزار پیکربندی پرینتر مبتنی بر وب را باز می‌کند. برخی از توزیع‌ها همچنین ابزارهای پیکربندی پرینتر خود را ارائه می‌دهند.

## Managing Drivers

بیشتر دستگاه‌های سخت‌افزاری نیازمند وجود اجزای نرم‌افزاری خاص برای مفید بودن هستند. یک قسمت از نرم‌افزاری که با سخت‌افزار "صحبت" می‌کند، به نام درایور معروف است، بنابراین شما باید بدانید چگونه درایورها در لینوکس کار می‌کنند. این بخش ابتدا به توصیف چندین دسته عظیم از درایورها که ممکن است بخواهید در محیط لینوکس خود استفاده کنید، می‌پردازد، سپس به بررسی چگونگی یافتن و نصب درایورها برای سخت‌افزارهای خود می‌پردازد.

## Understanding Types of Drivers

لینوکس به درایورها نیاز دارد زیرا سخت‌افزارهای مختلف حتی دو دستگاه که هدفهای بسیار مشابهی دارند، مانند دو آداتپور شبکه، می‌توانند به روش‌های بسیار متفاوتی عمل کنند. به عبارت دیگر، روش‌های مورد نیاز برای راه اندازی اولیه و استفاده از دو آداتپور شبکه ممکن است کاملاً متفاوت باشند. برای فراهم کردن رابطه‌ای سراسری به گونه‌ای که برنامه‌هایی مانند مرورگر وب Firefox بتوانند از هر آداتپور شبکه استفاده کنند، کرنل لینوکس از درایورها به عنوان پل بین رابطه‌های کرنل بدون وابستگی به سخت‌افزار و خود سخت‌افزار استفاده می‌کند.

نکته: در واقع، چندین لایه بین سخت‌افزار شبکه و یک برنامه مانند Firefox وجود دارد؛ درایور تنها یکی از این لایه‌ها است.

بطور کلی، درایورها می‌توانند در یکی از دو مکان وجود داشته باشند:

- داخل کرنل
- در یک کتابخانه یا برنامه خارجی

بیشتر درایورها مبتنی بر کرنل هستند و در واقع بخش بزرگی از کرنل لینوکس از درایورها تشکیل شده است. درایورهای کرنل بیشتر دستگاههای داخلی کامپیوتر را کنترل می‌کنند، مانند هارد دیسکها، رابطهای شبکه و رابطهای USB. دلیل اصلی اینکه بیشتر درایورها در کرنل قرار دارند این است که درایورها به طور معمول نیاز به دسترسی به سخت‌افزار دارند و این دقیقاً هدف کرنل است.

برخی از درایورها در کتابخانه یا برنامه خارج از کرنل واقع شده‌اند. به عنوانیں زیر:

- که اسکنرها را کنترل می‌کند SANE
- که خروجی چاپ را به یک فرم تبدیل می‌کند که پرینترهای خاصی می‌توانند آن را درکنند Ghostscript
- X که نمایشگر را مدیریت می‌کند

X از این لحاظ غیرمعمول است یک عنصر غیر از کرنل است که به نوعی به صورت مستقیم با سخت‌افزار گرافیک ارتباط برقرار می‌کند. در مقابل، SANE و Ghostscript هر دو از طریق رابطهای (مانند یک پورت USB) که توسط کرنل مدیریت می‌شوند، با دستگاههای سخت‌افزار خارجی ارتباط برقرار می‌کنند. به عبارت دیگر، برای مدیریت چنین دستگاههایی حداقل نیاز به دو درایور است. به عنوان مثال، برای چاپ روی یک پرینتر USB، از درایور USB کرنل و یک درایور پرینتر Ghostscript استفاده می‌شود. بهتر است اکثر کاربران از پیچیدگی این موارد آگاه نشوند، اما ممکن است شما بخواهید با آن آشنا شوید در صورتی که مشکلات پیش آید.

## Locating and Installing Drivers

بیشترین درایورها با خود کرنل لینوکس یا با کتابخانه یا برنامه‌ای که نوع سخت‌افزار را مدیریت می‌کند، همراه هستند. به عنوان مثال، بیشتر نصب‌های X شامل یک مجموعه از درایورهای گرافیک هستند تا بتوانند از اکثر کارت‌های گرافیک استفاده کنند. به همین دلیل، به ندرت لازم است تا به دنبال یافتن و نصب درایورهای اضافی برای سخت‌افزارهای متداول بروید. البته استثنایاتی وجود دارد، مانند:

**سخت‌افزار جدید:** اگر از سخت‌افزار شما بسیار جدید باشد (یعنی مدل آن جدید باشد، نه فقط اینکه به تازگی خریداری شده باشد)، ممکن است به درایورهایی نیاز داشته باشد که هنوز در توزیعی که استفاده می‌کنید، قرار نگرفته باشند.

**سخت‌افزارهای غیرمعمول:** اگر از سخت‌افزارهای عجیب مثل یک برد جمع‌آوری داده‌های علمی ویژه استفاده می‌کنید، ممکن است نیاز به یافتن درایور برای آن داشته باشید.

درایورهای انحصاری: برخی از تولیدکنندگان درایورهای انحصاری برای سخت‌افزارهای ایشان فراهم می‌کنند. به عنوان مثال، درایورهای گرافیک nvidia و fglrx (که در قسمت "Using Common Display Hardware" به آن اشاره شده است) می‌توانند عملکرد نمایشگرهای گرافیک مبتنی بر چیپست‌های NVIDIA یا ATI/AMD را بهبود بخشند. برخی از سخت‌افزارها به درایورهای انحصاری نیاز دارند. این امر به ویژه برای برخی از سخت‌افزارهای غیرمعمول رایج است.

اصلاحات باگ: درایورها، مانند سایر نرم‌افزارها، ممکن است دچار باگ شوند. اگر با چنین مشکلی روبرو شدید، ممکن است بخواهید یک درایور جدیدتر را پیدا کنید تا بهبود باگ را دریافت کنید. یک راه برای به‌دست آوردن یک درایور جدید مبتنی بر کرنل، ارتقاء کرنل است. توجه داشته باشید که ارتقاء کرنل همچنین می‌تواند هم اصلاحات باگ به درایورهای موجود و هم درایورهای کاملاً جدید ارائه دهد. به همین ترتیب، ارتقاء نرم‌افزار مانند Ghostscript یا SANE یا X می‌تواند درایورها را برای دستگاه‌هایی که چنین بسته‌هایی را اداره می‌کنند، ارتقا یا اضافه کند.

اگر از سخت‌افزار غیرمعمولی استفاده می‌کنید یا به درایوری دیگری که به دست آوردن آن دشوار است نیاز دارید، کار شما ممکن است دشوارتر باشد. می‌توانید با تولیدکننده تماس بگیرید یا یک سرج وب انجام دهید تا سعی کنید درایورها را پیدا کنید.

اگر یک درایور به عنوان بخشی از کرنل (یا بسته‌نرم‌افزار برای مدیریت دستگاه) در دسترس نباشد، باید دستورالعمل‌هایی که با درایور همراه است را مطالعه کنید. روش‌های نصب از یک درایور به درایور دیگر بسیار متفاوت است، بنابراین امکان ارائه یک روش نصب ساده و گام به گام که در همه موارد کار کند وجود ندارد. برخی از درایورها دارای ابزارهای نصب هستند، اما برخی دیگر شما را مجبور به دنبال یک روش می‌کنند که شامل تایپ دستورات مختلف است. اگر خوش شانس باشید، درایور به شکل یک پچ کرنل ممکن است به دست شما برسد. این یک راه برای اضافه یا تغییر فایل‌ها در پکیج سورس کد اصلی کرنل است. در این صورت باید کرنل را دوباره کامپایل کنید؛ کاری که خارج از دامنه این کتاب است.

# Chapter 7

## Managing Files

بسیاری از کارهایی که با یک کامپیوتر انجام می‌دهید، شامل دستکاری فایل‌هاست. فایل‌ها نامه‌ها، عکس‌های دیجیتال و سندهای دیگری را که ایجاد می‌کنید، نگه می‌دارند. همچنین، فایل‌ها تنظیمات پیکربندی لینوکس را نیز در اختیار دارند؛ مانند اطلاعاتی در مورد اینکه چگونه باید با اینترفیس‌های شبکه رفتار کرد، چگونه به هارد دیسک‌ها دسترسی داشته باشید و چه کارهایی هنگام استارت آپ کامپیوتر انجام می‌دهید. به واقع، حتی دسترسی به اکثر دستگاه‌های سخت‌افزاری و تنظیمات کرنل نهایتاً از طریق فایل‌ها انجام می‌شود. اطلاع داشتن از اینکه فایل‌ها را کجا پیدا کنید و چگونه آنها را مدیریت کنید، برای مدیریت یک کامپیوتر با لینوکس بسیار مهم است. این فصل با توضیحی از ساختار اولیه جایی که لینوکس فایل‌ها را ذخیره می‌کند شروع می‌شود. سپس به شما نشان می‌دهد چگونه در سیستم فایل لینوکس حرکت کنید تا به فایل‌های خود دسترسی پیدا کنید. در نهایت، دستورات پایه حالت متنی برای دستکاری فایل‌ها و دایرکتوری‌ها را مرور می‌کند.

### Understanding Where Things Go

همانطور که در فصل 6، "Managing Hardware" بحث شد، لینوکس از یک درخت دایرکتوری یکپارچه استفاده می‌کند؛ به عبارت دیگر، هر پارتیشن، دیسک قابل حمل، اشتراک‌گذاری فایل شبکه، و سایر دستگاه‌ها یا دستگاه‌های ذخیره ساز دیسک مانند، به عنوان یک دایرکتوری در یک درخت دایرکتوری واحد (یا فایل سیستم) قابل دسترسی است. این فایل سیستم ساختار یافته است؛ برخی از دایرکتوری‌ها هدف خاصی دارند، این دایرکتوری‌ها یا به عنوان زیردایرکتوری‌های منظم در یک پارتیشن وجود دارند و یا از روی دستگاه‌های جداگانه‌ای از root (/) مانند شده‌اند. درک هدف دایرکتوری‌های اصلی به شما کمک خواهد کرد تا فایل‌ها را پیدا کنید و از انجام اشتباهات نابخردانه جلوگیری کنید. با این حال، قبل از وارد شدن به جزئیات این مسائل، باید تفاوت بین فایل‌های کاربر و فایل‌های سیستم را بفهمید.

نکته: عبارت فایل سیستم ممکن است به ساختار داده‌های سطح پایین یا به سازماندهی فایل‌ها و دایرکتوری‌ها در کامپیوتر اشاره داشته باشد. در این فصل، اغلب اشاره به معنای دوم صورت می‌گیرد که به سازماندهی فایل‌ها و دایرکتوری‌ها در کامپیوتر اشاره دارد.

### User Files vs. System Files

برای درک تفاوت بین فایل‌های کاربر و فایل‌های سیستم، به یاد بیارید که لینوکس یک سیستم عامل چند کاربره است. در اصول، یک کامپیوتر تنها می‌تواند میزبان هزاران کاربر باشد. یک لحظه به چنین کامپیوتری فکر کنید؛ شاید یک مین فریم در یک دانشگاه، یک شرکت بزرگ یا یک سرور رایانش ابری باشد. اکثر این کاربران احتمالاً با جزئیات مدیریت سیستم لینوکس آشنا نخواهند بود؛ آن‌ها تنها می‌خواهند از پردازشگر کلمات، کلاینت‌های ایمیل و سایر برنامه‌ها استفاده کنند. این کاربران نیازی به سر و کله با فایل‌های پیکربندی سیستم ندارند. به واقعیت این است که اعطای دسترسی به این فایل‌ها به این کاربران، به خصوص

دسترسی به ایجاد تغییر، ممکن است بسیار خطرناک باشد. در یک کامپیوتر با 1,000 کاربر، هر کدام از آنها که ممکن است تنظیمات سیستم را تغییر دهد، می‌تواند باعث ایجاد تغییراتی شود که ممکن است کل کامپیوتر را به واسطه عدم آگاهی یا عمدی منهدم کند.

البته، مسائل مرتبط با محافظت از کامپیوتر از کاربرانش تنها یک حالت خاص از مسائل کلی مربوط به حساب‌های کاربری هستند؛ در آن کامپیوتر با 1000 کاربر، احتمالاً نمی‌خواهید کاربران بتوانند به جزء شیوه‌های محدود، فایل‌های یکدیگر را بخوانند و بنویسند. بنابراین، همانطور که در فصل 13 "Creating Users and Groups" و فصل 14 "Setting Ownership and Permissions" توضیح داده شده است، می‌توانید مجوزهایی را بر روی فایل‌ها تنظیم کنید تا از دسترسی غیرمجاز جلوگیری کنید.

فایل‌های سیستم، فایل‌هایی هستند که نحوه عملکرد کامپیوتر را کنترل می‌کنند. این شامل موارد زیر است:

- اسکریپت‌های شروع سیستم که سرورها و دیمون‌های مهم دیگر را راهاندازی می‌کنند.
- فایل‌های برنامه، هم فایل‌های باینری و هم اسکریپت‌ها.
- فایل‌های پشتیبانی برنامه مانند فونت‌ها و آیکون‌ها.
- فایل‌های پیکربندی که تعیین می‌کنند چگونه سیستم عمل می‌کند (تنظیمات پیکربندی شبکه، اطلاعات ترتیب دیسک و غیره).
- فایل‌های پیکربندی برای اکثر خدمات سرور و سایر دیمون‌ها.
- ذخیره داده برای برنامه‌های سیستم، مانند دیتابیس که توضیح می‌دهد چه برنامه‌هایی نصب شده‌اند.
- فایل‌های لاگ سیستم، که فعالیت‌های عادی سیستم را ثبت می‌کنند.

قاععدتاً، کاربران غیرتکنیکی نباید قادر به تغییر فایل‌های سیستم باشند، شاید حداقل به صورت غیرمستقیم. (برای مثال، فایل‌های لاگ فعالیت‌هایی مانند تلاش‌های لاجین به سیستم را ثبت می‌کنند.) کاربران باید قادر به خواندن برخی از انواع فایل‌های سیستم باشند، مانند فونت‌ها و آیکون‌هایی که استفاده می‌کنند، اما برخی از فایل‌های سیستم باید حتی از دسترسی خواندن هم محافظت شوند. (برای مثال، کاربران نباید قادر به خواندن فایل `/etc/shadow` باشند زیرا این فایل حاوی رمزهای رمزگاری شده است).

برای دستیابی به هدف محدود کردن دسترسی کاربران عادی به فایل‌های سیستم، این فایل‌ها به طور معمول متعلق به `root` یا به یک حساب دیگر سیستم هستند که منظور محدودتری دارد. به عنوان مثال، بسیاری از برنامه‌های سرور به سیستم اکانت‌های خود تکیه می‌کنند. فایل‌های سیستم سپس می‌توانند با تنظیم دسترسی‌ها به یک شیوه مناسب بسته به نیازهای خاص خود، محافظت شوند. بنابراین کاربران عادی قادر به ایجاد تغییر در اکثر فایل‌های سیستم نیستند و این فایل‌ها از آسیب محافظت می‌شوند. از آنجایی که `root` قادر به خواندن و نوشتن هر فایلی است، برای انجام اکثر وظایف نگهداری سیستم باید ابتدا از امتیازهای `root` استفاده کنید.

دقیقاً پس از نصب لینوکس، اکثر فایل‌هایی که وجود دارند، فایل‌های سیستم هستند و اکثر دایرکتوری‌ها و زیردایرکتوری‌ها در یک نصب تازه لینوکس، دایرکتوری‌های سیستم هستند. همانطور که به زودی توضیح داده می‌شود، چندین دایرکتوری مانند `/tmp` و `/home` برای فایل‌های کاربر تعیین شده‌اند، با این حال حتی این

دایرکتوری‌ها نیز به گونه‌ای ساختاردهی یا پیکربندی شده‌اند تا مشکلات دسترسی چند کاربره را جلوگیری کنند.

نکته: دایرکتوری‌های خانه کاربران معمولاً در `/home` قرار دارند، در حالی که `/tmp` برای همه کاربران قابل دسترس است و فایل‌های موقت را نگهداری می‌کند.

تفاوت بین فایل‌های سیستم و فایل‌های کاربر حتی در کامپیوترهای تک‌کاربری لینوکس، مانند لپتاپ شخصی شما، وجود دارد. این ممکن است عجیب یا حتی ناراحت‌کننده به نظر برسد؛ در نهایت، اگر شما تنها کاربر هستید و دسترسی روت دارید، چرا با تفاوت بین فایل‌های سیستم و فایل‌های کاربر اذیت شوید؟ پاسخ این است که این یک لایه حفاظتی در برابر آسیب اتفاقی یا خرابکارانه فراهم می‌کند. اگر یک اشتباه تایپ، یک باگ یا بدافزار، به عنوان مثال، تمام فایل‌های کامپیوتر را پاک کند، آسیب در صورت اجرای این عمل به عنوان یک کاربر عادی تا حدودی محدود می‌شود و گرنه یک کاربر عادی نمی‌تواند تمام فایل‌های کامپیوتر را حذف کند. بنابراین، تفاوت بین این دو نوع اکانت (و در نتیجه این دو کلاس فایل) حتی در یک کامپیوتر تک‌کاربری مفید است.

## The Filesystem Hierarchy Standard

هر چند هر توزیع لینوکسی راه منحصر به فردی برای انجام برخی از کارها دارد، اما توسعه‌دهندگان همه آنها نیاز به استانداردسازی در چیدمان دایرکتوری‌هایشان دارند. به عنوان مثال، برنامه‌ها باید بتوانند فایل‌های پیکربندی سیستمی کلیدی را به صورت یکنواخت در همه توزیع‌ها در جاهای یکسان پیدا کنند. اگر اینطور نبود، برنامه‌هایی که بر این ویژگی‌ها وابسته هستند، پیچیده‌تر می‌شوند و ممکن بود که در همه توزیع‌ها کار نکنند. برای پاسخ به این نیاز، استاندارد سلسله مراتب فایل‌ها (FHS) ایجاد شد. علاوه بر لینوکس، برخی از سیستم‌عامل‌های مشابه یونیکس نیز تا حدودی از FHS پیروی می‌کنند.

نکته: از یک استاندارد قبلی که فقط مختص لینوکس بود، یعنی استاندارد سلسله مراتب فایل (FSSTND)، به وجود آمد.

یک تفاوت مهم توسط FHS بین فایل‌های قابل به اشتراک‌گذاری و فایل‌های غیرقابل به اشتراک‌گذاری است. فایل‌های قابل به اشتراک‌گذاری، مانند فایل‌های داده کاربر و فایل‌های اجرایی برنامه، می‌توانند به طور معقولی بین کامپیوترها به اشتراک گذاشته شوند. (به طور طبیعی، نیازی به به اشتراک گذاری چنین فایل‌هایی ندارید، اما ممکن است این کار را انجام دهید). اگر فایل‌ها به اشتراک گذاشته شوند، معمولاً از طریق یک سرور سیستم فایل شبکه (NFS) به اشتراک گذاشته می‌شوند. فایل‌های غیرقابل به اشتراک‌گذاری شامل اطلاعات وابسته به سیستم هستند، مانند فایل‌های پیکربندی. به عنوان مثال، شما احتمالاً نخواهید یک فایل پیکربندی سرور را بین کامپیوترها به اشتراک بگذارید.

تفاوت دیگری که توسط FHS معین می‌شود، تفاوت بین فایل‌های استاتیک و فایل‌های متغیر است. استاتیک معمولاً به جز از طریق مداخله مستقیم مدیر سیستم تغییر نمی‌کنند. بیشتر برنامه‌های اجرایی نمونه‌های خوبی از فایل‌های استاتیک هستند. فایل‌های متغیر توسط کاربران، اسکریپت‌های خودکار، سرورها یا موارد

مشابه تغییر می‌کنند. به عنوان مثال، دایرکتوری‌های home کاربران و صفحه‌ای ایمیل از فایل‌های متغیر تشکیل شده‌اند.

FHS سعی می‌کند هر دایرکتوری را در یک خانه از این ماتریس  $2 \times 2$  (قابل به اشتراک/غیرقابل به اشتراک استاتیک/متغیر) جدا کند. جدول 7.1 این ارتباطات را نشان می‌دهد. برخی از دایرکتوری‌ها زیردایرکتوری‌های را در چندین خانه شامل می‌شوند، اما در این موارد، FHS سعی می‌کند وضعیت زیردایرکتورهای خاص را مشخص کند. به عنوان مثال، /var/متغیر است و شامل برخی زیردایرکتورهای قابل به اشتراک و برخی زیردایرکتورهای غیرقابل به اشتراک است، همانطور که در جدول 7.1 نشان داده شده است.

Table 7.1 The FHS directory classification system

	Shareable	Unshareable
Static	/usr, /opt	/etc, /root
Variable	/Home, /var/mail	/var/run, /var/lock

به صورت نسخه‌های شماره‌گذاری شده ارائه می‌شود. نسخه 3.0، آخرین نسخه تا زمان نوشتن این متن، در ماه مه 2015 منتشر شد (برای اطلاعات بیشتر به صفحه وب FHS در [linuxfoundation.org/collaborate/workgroups/lsb/fhs](http://linuxfoundation.org/collaborate/workgroups/lsb/fhs) مراجعه کنید).

## Important Directories and Their Contents

FHS نام‌ها و اهداف بسیاری از دایرکتوری‌ها و زیردایرکتوری‌ها را در یک سیستم لینوکس تعریف می‌کند. جدول 7.2 خلاصه‌ای از مهمترین این دایرکتوری‌ها را ارائه می‌دهد. بیشتر این دایرکتوری‌ها دایرکتوری‌های سیستم هستند، با استثنای اصلی‌ها مانند /tmp، /mnt، /home و ./media.

Table 7.2 Important Linux directories according to the FHS

Directory	Purpose
/	دایرکتوری روت. همه فایل‌ها در این دایرکتوری یا زیردایرکتوری‌های آن قرار دارند.
/etc	فایل‌های پیکربندی سیستم در این دایرکتوری قرار دارند.
/boot	دارای فایل‌های بوت مهم مانند کرنل لینوکس، دیسک حافظه RAM اولیه (initrd)، و اغلب فایل‌های پیکربندی بوت لودر است.

/bin	دارای فایل‌های برنامه که برای عملکرد عادی حیاتی هستند و کاربران عادی ممکن است آن‌ها را اجرا کنند.
/sbin	دارای فایل‌های برنامه که برای عملکرد عادی حیاتی هستند و کاربران عادی به ندرت آن‌ها را اجرا می‌کنند.
/lib	دارای کتابخانه‌ها - کدی که توسط بسیاری از برنامه‌های دیگر استفاده می‌شود که برای عملکرد اساسی سیستم حیاتی هستند.
/usr	شامل برنامه‌ها و داده‌های استفاده شده در عملکرد عادی سیستم است، اما برای بوت ابتدایی سیستم ضروری نیست. این دایرکتوری به زیردایرکتوری‌هایی تقسیم شده است که ساختار روت را تکرار می‌کنند - /usr/bin، /usr/sbin، /usr/lib/ و غیره.
/home	شامل دایرکتوری‌های خانه کاربران است. جدا کردن این دایرکتوری به عنوان یک سیستم فایل سطح پایین به طور موثر اطلاعات بسیاری از کاربر را از سیستم عامل جدا می‌کند، که ممکن است مفید باشد اگر بخواهید سیستم عامل را مجدداً نصب کنید بدون از دست دادن اطلاعات کاربر.
/root	دایرکتوری خانه root.
/var	دایرکتوری متفرقه برای فایل‌های موقتی مانند فایل‌های لگ و فایل‌های پرینت اسپول. یک زیردایرکتوری ویژه از /var/ به نام /var/tmp وجود دارد. مانند /tmp (که در ادامه توضیح داده خواهد شد)، /var/tmp فایل‌های موقت را نگه می‌دارد. این فایل‌ها نباید هنگام ریبوت کامپیوتر حذف شوند.
/tmp	نگهداری فایل‌های موقت، اغلب شامل فایل‌های موقتی است که توسط برنامه‌های کاربر ایجاد شده‌اند. در تئوری، این فایل‌ها ممکن است هنگام ریبوت کامپیوتر حذف شوند، اگرچه در عمل بسیاری از توزیع‌ها این کار را انجام نمی‌دهند.
/mnt	محل اتصال معمول برای رسانه‌های قابل حمل؛ گاهی

	اوقات به زیرشاخه‌های مختلف برای هر سیستم فایل متصل شده تقسیم می‌شود.
/media	محل اتصال جدید برای رسانه‌های قابل حمل؛ به طور معمول به زیرشاخه‌های مختلف برای هر سیستم فایل متصل شده تقسیم می‌شود.
/dev & /run	نگهداری فایل‌های دستگاه که دسترسی سطح پایین به سخت‌افزار را فراهم می‌کنند اطلاعات درباره سیستم در حال اجرا

به عنوان یک کاربر عادی، بیشتر فایل‌های خود را در دایرکتوری خانگی‌تان ایجاد خواهید کرد که معمولاً یک زیردایرکتوری از /home است. شما همچنین ممکن است به منابع جابجایی‌پذیر که در /media (یا گاهی /run/media) مانند شده باشند، و شاید به منابع شبکه که ممکن است در جاهای دیگری مانند شوند، دسترسی داشته باشید. همچنین می‌توانید از /tmp و برخی زیردایرکتوری‌های خاص /var استفاده کنید، اگرچه بیشتر کاربران نیازی به آگاهی صریح از این مکان‌ها ندارند؛ برنامه‌ها به طور معمول برنامه‌ریزی شده‌اند تا از آنها برای فایل‌های موقت یا انواع خاصی از فایل‌ها، مانند فایل‌های دریافتی ایمیل، استفاده کنند. به عنوان یک مدیر سیستم، ممکن است فایل‌های قرار گرفته در هر کدام از این دایرکدام را تغییر دهید؛ اما برای یک مدیر سیستم، /etc به ویژه مهم است، زیرا بیشتر فایل‌های پیکربندی سیستم در آنجا قرار دارند. هنگامی که با ابزارهای گرافیکی یا متنی، سیستم خود را کاوش می‌کنید، می‌بایست به این ساختار دایرکتوری توجه داشته باشید.

**نکته:** کاربران عادی قادر به نوشتن در اکثر دایرکتوری‌های سیستم، مانند /usr، نیستند. بنابراین، شما نمی‌توانید با بررسی این دایرکتوری‌ها به سیستم عامل خود آسیب برسانید. به عبارت دیگر، اگر به عنوان یک کاربر عادی وارد شوید، نمی‌توانید سیستم عامل خود را خراب کنید!

از این دایرکتوری‌ها، تعدادی از دایرکتوری‌ها یا مجموعه‌های آنها نیاز به توجه خاص دارند:

**دایرکتوری پیکربندی:** دایرکتوری /etc بیشترین تعداد فایل‌های پیکربندی سیستم را نگه می‌دارد. فصل‌های گذشته به چندین فایل چنینی اشاره کردند، مانند /etc/fstab (که محل مانند کردن پارتیشن‌ها را تعریف می‌کند) و /etc/passwd (که فایل اصلی تعریف اکانت است). بسیاری دیگر وجود دارد. در واقع، شما زیردایرکتوری‌هایی در /etc برای نگهداری فایل‌های پیکربندی چندین ساب سیستم و سرور پیچیده مانند /etc/samba (برای سرور فایل Samba) و /etc/X11 (برای X Window System) وجود دارند.

**دایرکتوری‌های قابل اجرا:** فایل‌های برنامه اصلی اصولاً در /bin، /sbin، /usr/bin و /usr/sbin قرار دارند. (دایرکتوری‌های اضافی برخی از سیستم‌ها ممکن است فایل‌های برنامه را در خود جای دهند. به ویژه، برنامه‌هایی که به صورت لوکال کامپایل شده‌اند را نگهداری می‌کنند.)

**دایرکتوری‌های کتابخانه:** کتابخانه‌ها مجموعه‌هایی از توابع برنامه‌نویسی هستند که می‌توانند برای بسیاری از برنامه‌ها مفید باشند. آن‌ها در فایل‌های جداگانه ذخیره می‌شوند تا در زمان اجرای برنامه‌ها، فضای دیسک و RAM را صرفه‌جویی کنند و امکان بروزرسانی آسان فایل‌های کتابخانه بدون نیاز به نصب مجدد تمام برنامه‌هایی که به آن‌ها وابسته‌اند را فراهم کنند. در لینوکس، بیشتر کتابخانه‌ها در /lib و /usr/lib قرار دارند، هرچند برخی از آن‌ها ممکن است در سیستم‌های مانند /usr/local/lib قرار گیرند.

اگر با کامپیوترهای ویندوزی کار کرده باشید، باید از یک تفاوت مهم بین ویندوز و لینوکس آگاه باشید: در ویندوز، معمول است که یک برنامه اجرایی، فایل‌های پیکربندی و تمام فایل‌های پشتیبانی آن در یک درخت دایرکتوری ویژه برنامه قرار داشته باشد، مانند C:\Program Files\SomeProgram. با این حال، در لینوکس، بیشتر فایل‌های کلیدی یک برنامه احتمالاً در مکان‌های استاندارد قرار دارند که با سایر برنامه‌ها به اشتراک گذاشته می‌شوند و پراکنده هستند. به عنوان مثال، فایل اجرایی برنامه ممکن است در /usr/bin باشد، کتابخانه‌های مرتبط در /usr/lib، فایل‌های پیکربندی در /etc/ یا در دایرکتوری‌های خانه کاربران، وغیره. این سیستم در لینوکس عالی عمل می‌کند زیرا سیستم‌های پکیجینگ لینوکس، که در فصل 9 "Exploring Processes and Process Data" توضیح داده شده‌اند، موقعیت فایل‌های یک پکیج را ثبت می‌کنند و به شما امکان حذف یا ارتقاء یک پکیج را به سهولت می‌دهند. لینوکس این امتیاز را دارد که مسیر (لیست دایرکتوری‌هایی که فایل‌های برنامه در آنها قرار دارند) را ساده‌تر می‌کند. (مسیرها همچنین برای کتابخانه‌ها و صفحات man وجود دارند). اگر شما عادت به گشتن به دنبال فایل‌ها در مکان‌های خاص برنامه دارید، استفاده از سیستم لینوکس می‌تواند کمی دشوار باشد؛ کلید، استفاده از سیستم پکیجینگ خود است تا محل فایل‌های یک بسته را شناسایی کنید؛ به عنوان مثال، تایپ کردن `rpm -ql someprogram` نشان می‌دهد که هر فایل در پکیج someprogram در یک سیستم مبتنی بر RPM در کجا قرار دارد.

**نکته:** سیستم پکیجینگ فایل‌های پیکربندی کاربر را مدیریت نمی‌کند، بنابراین این فایل‌ها ممکن است پس از حذف یک برنامه باقی بمانند. این هیچ آسیبی به جز مصرف فضای دیسک ایجاد نمی‌کند.

## Exploring Files and Directories

وقتی با ساختار فایل‌های لینوکس آشنا شدید، احتمالاً آماده کمی کاوش هستید. در صفحات بعدی، توضیح می‌دهیم چگونه بفهمید چه فایل‌هایی در هارد دیسک شما وجود دارد، چگونه بین دایرکتوری‌ها حرکت کنید، چگونه به فایل‌هایی ارجاع دهید که در دایرکتوری فعلی نیستند، و چگونه از پرکاربردترین دستورات برای اداره فایل‌ها استفاده کنید.

## Obtaining File Listings

برای اداره فایل‌ها، مفید است بدانید چه فایل‌هایی وجود دارند. دستور ls، که نام کوتاه list است، این اطلاعات را به شما ارائه می‌دهد. دستور ls نام فایل‌ها را در یک دایرکتوری نمایش می‌دهد. اگر هیچ گزینه‌ای به آن ندهید، فایل‌ها را در دایرکتوری فعلی نشان می‌دهد؛ با این حال، می‌توانید به آن گزینه‌ها به همراه مشخصات فایل یا دایرکتوری بدهید. این دستور از تعداد زیادی از گزینه‌ها پشتیبانی می‌کند؛ برای جزئیات

بیشتر می‌توانید به صفحه man آن مراجعه کنید. جدول 7.3 خلاصه‌ای از مهمترین گزینه‌های ls را نشان می‌دهد.

Table 7.3 Common ls options

Option (long form)	Option (short form)	Description
--all	-a	نمایش فایل‌های نقطه‌ای. به طور معمول، دستور ls از نمایش فایل‌هایی که با نقطه (.) شروع می‌شوند خودداری می‌کند. این فایل‌های نقطه‌ای (همچنین به عنوان فایل‌های پنهان شناخته می‌شوند) اغلب فایل‌های پیکربندی هستند که معمولاً جذابیت زیادی ندارند.
--color	N/A	تولید یک لیست با رنگ کد شده که از طریق نمایش فایل‌ها و نوع‌های خاص دیگر را با رنگ‌های مختلف تمایز می‌دهد. برخی از توزیع‌های لینوکس شل‌های خود را به گونه‌ای پیکربندی می‌کنند که از این گزینه به صورت پیش‌فرض استفاده می‌کنند.
--directory	-d	تغییر رفتار ls به گونه‌ای که فقط نام دایرکتوری را لیست کند. به طور معمول، اگر نام یک دایرکتوری، را به عنوان یک گزینه وارد کنید، ls محتوای آن دایرکتوری را نمایش می‌دهد. همین اتفاق تکرار می‌شود اگر نام یک دایرکتوری با یک wildcard مطابقت داشته باشد.
N/A	-l	تولید یک لیست طولانی که شامل

		اطلاعاتی مانند رشته مجوز فایل، مالک، گروه، اندازه و تاریخ ایجاد فایل می‌شود. دستور <code>ls</code> به طور معمول فقط نام فایل‌ها را نمایش می‌دهد.
--file-type	-F	افزودن یک کد نشانگر به انتهای هر نام تا بدانید چه نوعی از فایل است.
--recursive	-R	موجب می‌شود <code>ls</code> محتوای دایرکتوری را به صورت بازگشتنی نمایش دهد. به عبارت دیگر، اگر دایرکتوری هدف حاوی یک زیردایرکتوری باشد، <code>ls</code> همزمان فایل‌های دایرکتوری هدف و فایل‌های زیردایرکتوری را نمایش می‌دهد. نتیجه ممکن است یک لیست بزرگ باشد اگر یک دایرکتوری شامل تعداد زیادی زیردایرکتوری باشد.

شما می‌توانید به اختیار به `ls` یک یا چند نام فایل یا دایرکتوری بدهید، در این صورت `ls` اطلاعات مربوط به آن فایل یا دایرکتوری را نمایش می‌دهد، همانند این مثال:

```
$ ls -F /usr /bin/ls
/bin/ls
/usr:
X11R6/      games/          include/        man/         src/
bin/         i386-glibc20-linux/    lib/          merge@       tmp@
doc/         i486-linux-libc5/     libexec/      sbin/
etc/         i586-mandrake-linux/ local/        share/
```

این خروجی هم فایل برنامه `/bin/ls` و هم محتویات دایرکتوری `/usr` را نشان می‌دهد. این دایرکتوری اصلتاً از زیردایرکتوری‌هایی تشکیل شده است که با استفاده از خط کشیده `(/)` در انتهای وقتی که `-F`-استفاده می‌شود

نشان داده می‌شوند. به طور پیش‌فرض، ls یک لیست‌بندی ایجاد می‌کند که بر اساس نام فایل مرتب شده است، همانطور که در این مثال نشان داده شده است. توجه داشته باشید که حروف بزرگ (مانند 6X11R) همیشه قبل از حروف کوچک (مانند bin) ظاهر می‌شوند.

نکته: یک علامت at-sign (@) در انتهای، به یک لینک نمادین اشاره دارد که یک فایل به دیگری اشاره می‌کند. یکی از متداول‌ترین گزینه‌های -ls است که یک لیست طولانی از دایرکتوری می‌سازد مثال زیر:

```
$ ls -l t*
-rwxr-xr-x    1 rich rich          111 Aug 13 13:48 test
-rw-r--r--    1 rich rich      176322 Jul 16 09:34 thttpd-2.20b-
1.i686.rpm
-rw-r--r--    1 rich rich     1838045 Jul 24 18:52 tomsrtbt-
1.7.269.tar.gz
-rw-r--r--    1 rich rich     3265021 Aug 22 23:46 tripwire.rpm
```

این خروجی شامل رشته‌های مجوز (مانند rwxr-xr-x)، مالکیت (یک مالک به نام rich و یک گروه به نام rich برای تمام این فایل‌ها)، اندازه فایل‌ها و تاریخ ایجاد فایل است، علاوه بر نام‌های فایل. این مثال همچنین استفاده از وایلدکارد \* را نشان می‌دهد که هر رشته‌ای را مطابقت می‌دهد؛ بنابراین \*t هر نام فایلی که با t شروع شود را مطابقت می‌دهد.

نکته: فصل 14 تا حد زیادی به جزئیات موضوعات مالکیت و مجوز فایل می‌پردازد.

## Changing Directories

دستور cd دایرکتوری فعلی که در آن کار می‌کنید را تغییر می‌دهد. اگرچه برای بسیاری از دستورات، دایرکتوری فعلی مهم نیست، اما وقتی شروع به ارجاع به فایل‌ها می‌کنید، اهمیت پیدا می‌کند. همانطور که در بخش بعدی "Using Absolute and Relative File References" توضیح داده شده است، برخی از انواع ارجاع به فایل به دایرکتوری فعلی شما وابسته هستند.

وقتی که دایرکتوری فعلی خود را تغییر می‌دهید، ممکن است پرامپت شل شما (با توجه به تنظیمات توزیع شما) تغییر کند و به شکلی شبیه به این شود:

```
[rich@essentials ~]$ cd /usr/bin
[rich@essentials bin]$
```

نکته: در این کتاب، زمانی که دستورات را در خطوط مجزا نمایش می‌دهیم، بیشترین پرامپت‌های شل را به یک کاراکتر، مانند \$، کوتاه‌تر می‌کنیم.

پیش‌فرض برخی از تنظیمات توزیع‌های بسیاری تنها بخش آخر از دایرکتوری فعلی را نمایش می‌دهند، مثلًا bin به جای /usr/bin در مثال پیشین. اگر نیاز به دانستن مسیر کامل مکان فعلی خود دارید، می‌توانید از دستور pwd استفاده کنید:

```
$ pwd
```

```
/usr/bin
```

لینوکس از کاراکتر خط (/) به عنوان جداکننده دایرکتوری استفاده می‌کند. اگر با ویندوز آشنا هستید، ممکن است بدانید که ویندوز از علامت بک اسلش به عنوان جداکننده دایرکتوری استفاده می‌کند ()). این دو را اشتباہ نکنید! در لینوکس، یک بک اسلش به عنوان یک کاراکتر "نقل قول" یا "اسکیپ" عمل می‌کند تا بتوانید کاراکترهایی که در غیر این صورت سخت مشخص می‌شوند، مانند فاصله، به عنوان بخشی از یک نام فایل وارد کنید. همچنین توجه داشته باشید که بک اسلش به همین دلیل به عنوان یک کاراکتر قانونی در یک نام فایل لینوکس قابل استفاده نیست.

## Using Absolute and Relative File References

همانطور که در فصل 6، "Managing Hardware" توضیح داده شد، لینوکس از یک درخت دایرکتوری یکپارچه استفاده می‌کند، که به این معناست که همه فایل‌ها می‌توانند نسبت به یک دایرکتوری root که معمولاً با استفاده از کاراکتر خط (/) اشاره می‌شود، محل قرار گیرند. اگر دیسک شما شامل چندین پارتیشن باشد، یکی از این دستگاه‌ها به عنوان فایل سیستم root استفاده می‌شود و باقی در یک مکان دلخواه در داخل درخت دایرکتوری کلی مانند شده‌اند. همین اتفاق هنگامی رخ می‌دهد که یک فلاش درایو USB، دی‌وی‌دی، یا دستگاه دیسک قابل حمل دیگر مانند شود. نتیجه ممکن است شبیه به شکل 7.1 باشد که یک زیرمجموعه از دایرکتوری‌های موجود در یک لینوکس را به همراه چندین نوع رسانهٔ قابل انتقال نشان می‌دهد. بیشترین تعداد اوقات، رسانه‌های قابل انتقال به عنوان زیردایرکتوری‌های media/ ظاهر می‌شوند، اما برخی از توزیع‌های لینوکس ترجیح می‌دهند از run/media/ استفاده کنند. بیشتر زیردایرکتوری‌ها می‌توانند به عنوان پارتیشن‌های جداگانه جدا شوند یا حتی در دیسک‌های جداگانه قرار گیرند. در شکل 7.1، دایرکتوری /home در یک پارتیشن جداگانه قرار دارد، اما با دقت به همان روشی که اگر جزء پارتیشن root (/) بود ارتباط برقرار می‌شد، دسترسی دارد.

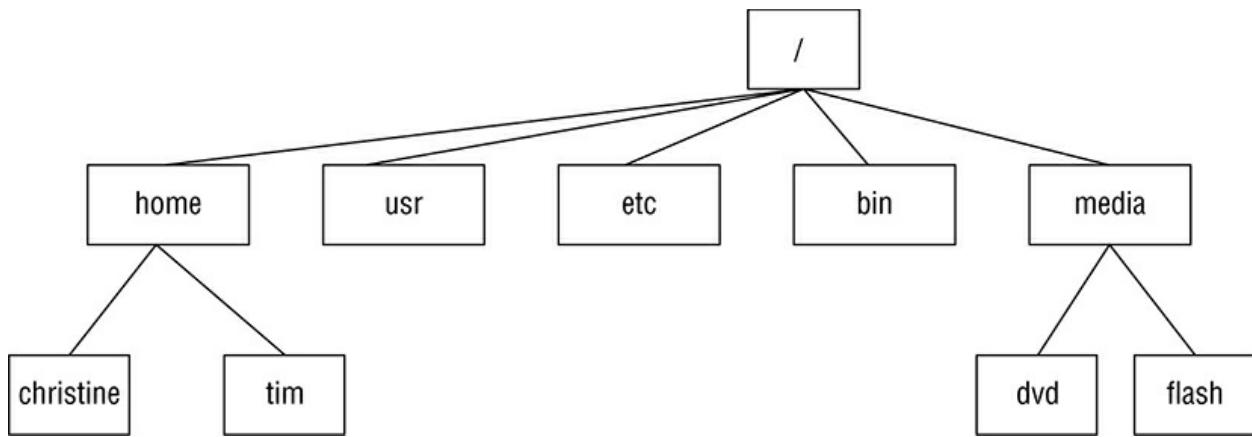


Figure 7.1 In Linux, all files are referred to relative to a single root (/) directory.

نکته: (/) را با دایرکتوری root /root /اشتباه نگیرید. root /دایرکتوری خانه کاربر root است.

هنگام اشاره به فایل‌ها و دایرکتوری‌ها در دستورات، می‌توانید به سه روش به آن‌ها ارجاع دهید:

**Absolute References**: ارجاعات مطلق به (/) دایرکتوری نسبت دارند، مثل /home/fred/afile.txt برای اشاره به فایل afile.txt در دایرکتوری خانه. چنین ارجاعاتی همیشه با یک کاراکتر خط (/) شروع می‌شوند.

**Home Directory References**: کاراکتر تیلدا (~) به دایرکتوری خانه کاربر اشاره دارد. اگر یک نام فایل با این کاراکتر شروع شود، به نظر می‌رسد که مسیر دایرکتوری خانه کاربر جایگزین شده است. بنابراین، برای fred، ~/home/fred/afile.txt معادل afile.txt / است.

**Relative References**: این نوع به دایرکتوری فعلی نسبت دارند. بنابراین، اگر فرد در دایرکتوری خانه‌اش کار می‌کند، /home/fred/afile.txt به afile.txt اشاره دارد. اشاره‌های نسبی می‌توانند شامل زیر دایرکتوری‌ها باشند، مانند somedir/anotherfile.txt در لینوکس، هر دایرکتوری شامل یک اشاره مخفی ویژه (..) است که به دایرکتوری بالا دستی اشاره دارد. بنابراین، اگر سالی در /home/sally در /kar می‌کند، می‌تواند به فایل afile.txt فرد به عنوان .. fred/afile.txt ارجاع دهد.

نکته: مجوزهای فایل می‌توانند به دلیل محدودیت‌های دسترسی، از دسترسی شما به فایل‌های کاربر دیگر جلوگیری کنند، که در فصل ۱۴ توضیح داده شده است.

برای بهتر درک این مفاهیم، این عملیات‌ها را امتحان کنید:

1. یک شل جدید اجرا کنید یا از یک شل موجود استفاده کنید.
2. ~ را تایپ کنید تا وارد دایرکتوری خانه‌تان شوید.
3. cat /etc/fstab را تایپ کنید تا این پرونده پیکربندی را با استفاده از یک ارجاع فایل مطلق ببینید.  
محتوا در ترمینال شما نمایش داده می‌شود.

4. `pwd` را تایپ کنید تا دایرکتوری فعلی خود را مشاهده کنید. احتمالاً `/home/yourusername/` خواهد بود، جایی که `yourusername`، نام کاربری شماست.

5. `cat` را تایپ کنید تا این پرونده پیکربندی را با استفاده از یک ارجاع فایل نسبی ببینید. اولین `..` در این دستور به `/home/` ارجاع دارد و دومین مورد به دایرکتوری `root` (`/`) ارجاع دارد. اگر دایرکتوری خانه شما در مکان غیر معمولی ای باشد، ممکن است نیاز به تنظیم تعداد `..` در این دستور داشته باشید؛ به همین دلیل شما را قادر به استفاده از `pwd` برای یافتن دایرکتوری فعلی (کردیم).

6. `cat` را تایپ کنید تا این پرونده پیکربندی را با استفاده از یک ارجاع دایرکتوری خانه ببینید.

البته، گام‌های 5 و 6 از ارجاع‌های نسبتاً گنگ استفاده می‌کنند؛ در واقعیت، شما احتمالاً از یک ارجاع فایل مطلق برای دسترسی به `/etc/fstab` از داخل دایرکتوری خانه‌تان استفاده می‌کنید. اگر در یک زیردایرکتوری از `/etc/` باشید، تایپ کردن `..` کمی ساده‌تر از تایپ `/etc/fstab` خواهد بود؛ و تایپ کردن `/etc/~` ساده‌تر از تایپ مسیر کامل به دایرکتوری خانه‌تان خواهد بود.

## Manipulating Files

اگر از ویندوز یا مک استفاده کرده‌اید، احتمالاً از یک فایل منیجر گرافیکی برای کنترل فایل‌ها استفاده کرده‌اید. چنین ابزارهایی نیز در لینوکس وجود دارند، همانطور که در فصل 4، "Using Common Linux Programs" ذکر شده است، و مطمئناً می‌توانید برای بسیاری از کارهای رایج از یک فایل منیجر استفاده کنید. شلهای متعدد در لینوکس، مانند Bash، ابزارهای ساده‌اما قدرتمندی را برای کنترل فایل‌ها فراهم می‌کنند. این ابزارها می‌توانند برخی از کارها را ساده‌تر کنند، مانند کار با تمامی فایل‌هایی که نامشان شامل رشته "invoice" است. بنابراین، با این دستورات متعدد باید آشنا باشید.

برای شروع این کار، راههایی را توضیح می‌دهیم که می‌توانید با آنها فایل ایجاد شده، می‌توانید آنها را از یک مکان به مکان دیگر کپی کنید. گاهی اوقات ممکن است بخواهید فایل‌ها را منتقل یا تغییر نام دهید، بنابراین توضیح می‌دهیم چگونه این کارها را انجام دهیم. لینوکس به شما امکان ایجاد لینک‌ها را می‌دهد، که راههایی برای ارجاع به همان فایل از طریق چندین نام هستند. اگر هرگز نخواهید از یک فایل استفاده کنید، می‌توانید آن را حذف کنید. ویژگی‌های خاص لینوکس در مورد حروف بزرگ و کوچک حروف در دستورات کنترل فایل‌ها را توضیح می‌دهیم.

## Creating Files

به طور معمول، شما فایل‌ها را با استفاده از برنامه‌هایی که آن‌ها را کنترل می‌کنند، ایجاد می‌کنید. به عنوان مثال، ممکن است از یک برنامه گرافیکی برای ایجاد یک فایل گرافیکی جدید استفاده کنید. این فرآیند در هر برنامه متفاوت است، اما برنامه‌های گرافیکی معمولاً از یک گزینه منو به نام "Save" یا "Save As" برای ذخیره یک فایل استفاده می‌کنند. برنامه‌های متعدد نیز امکانات مشابهی را فراهم می‌کنند، اما جزئیات اینکه این کار چگونه انجام می‌شود، از یک برنامه به برنامه دیگر بسیار متفاوت است.

نکته: فصل 10، "Editing Files" توضیحاتی در مورد ایجاد فایل‌های متنی با استفاده از ویراستارهای متنی nano و vi می‌دهد.

یک برنامه ویژه وجود دارد که به عنوان یک راه برای ایجاد فایل‌ها شناخته می‌شود: touch. شما می‌توانید نام این برنامه را وارد کنید و پس از آن نام یک فایل که می‌خواهید ایجاد کنید، مانند touch newfile.txt تا یک فایل خالی به نام newfile.txt ایجاد کنید. معمولاً از برنامه تخصصی ای برای انجام کار استفاده خواهد کرد. با این حال، گاهی اوقات، ممکن است مفید باشد یک فایل خالی ایجاد کنید فقط با خاطر داشتن خود فایل - به عنوان مثال، برای ایجاد چند "Scratch Files" برای تست یک دستور دیگر.

اگر به touch نام یک فایلی که قبلاً وجود دارد را بدهید، touch برچسب‌های زمان دسترسی و اصلاح آن فایل را به تاریخ و زمان فعلی بهروزرسانی می‌کند. این ممکن است مفید باشد اگر از یک دستور استفاده می‌کنید که بر اساس زمان‌های دسترسی به فایل‌ها کار می‌کند و می‌خواهید برنامه با یک فایل قدیمی به عنوان یک فایل جدید رفتار کند. همچنین شما ممکن است نیاز به انجام این کار را داشته باشید اگر قصد دارید یک مجموعه از فایل‌ها را توزیع کنید و می‌خواهید همه آن‌ها برچسب زمانی یکسان داشته باشند.

نکته: ابزاری که به نام make شناخته می‌شود، در صورتی که سورس کد جدید باشد، آن را کامپایل می‌کند، بنابراین برنامه‌نویسان گاهی از touch برای اجبار make به کامپایل مجدد یک فایل سورس کد استفاده می‌کنند.

شما می‌توانید از چندین آپشن با touch برای تغییر رفتار آن استفاده کنید. مهمترین این آپشن‌ها عبارتند از: فایلی ایجاد نکن: گزینه -c یا --no-create که اگر فایلی وجود نداشته باشد، فایل جدیدی ایجاد نکند. از این گزینه استفاده کنید تا هنگام بهروزرسانی برچسب‌های زمانی در صورتی که به اشتباه یک نام فایل را اشتباه وارد کردید، یک فایل خالی به طور اتفاقی ایجاد نشود.

زمان را به یک مقدار خاص تنظیم کنید: می‌توانید از گزینه -d یا --date="string" که می‌توانید تاریخ یک فایل را به تاریخ مشخص شده در رشته مشخص شده تنظیم کنید، که می‌تواند از هر تعداد اشکال باشد. به عنوان مثال، دستور touch -d "July 4 2019" afile.txt باعث تنظیم برچسب‌های زمانی روی afile.txt به تاریخ 4 ژوئیه 2019 می‌شود. می‌توانید این تأثیر را با دستور زیر نیز بدست آورید:

```
-t [[ CC ] YY] MMDDhhmm[. ss]
```

جایی که [[ CC ] YY] تاریخ و زمان را در یک فرمت عددی خاص نشان می‌دهد، مانند 201907041223 بعد از ظهر در 4 ژوئیه 2019.

صفحه راهنمای touch دستور شما را با گزینه‌های کمتر شناخته شده آن آشنا کند.

## Copying Files

اگر در یک شل متنی کار می‌کنید، دستور `cp` یک فایل را کپی می‌کند. (نام این دستور از `copy` گرفته شده است.) برای استفاده از آن، می‌توانید `cp` را با یک نام فایل مبدا و یک نام فایل مقصد، یک نام دایرکتوری مقصد، یا هر دو فراخوانی کنید. جدول 7.4 این سه روش استفاده از این دستور را شرح می‌دهد. هرچند که نامهای فایل مثال در جدول 7.4 نشان می‌دهد که فایل اصلی در دایرکتوری کنونی کاری شما است، این لزومی ندارد؛ فایل `orig.txt` می‌تواند شامل یک مشخصه دایرکتوری باشد، مانند `/etc/fstab` یا `./afile.txt`.

Table 7.4 Examples of the use of `cp`.

Effect	Example command
کپی کردن فایل <code>orig.txt</code> به نام <code>new.txt</code> در دایرکتوری کنونی.	<code>cp orig.txt new.txt</code>
کپی <code>orig.txt</code> به <code>new.txt</code> در دایرکتوری فعلی. کپی <code>orig.txt</code> به دایرکتوری <code>otherdir/otherdir</code> . کپی با نام <code>orig.txt</code> خواهد بود.	<code>cp orig.txt /otherdir/otherdir</code>
کپی <code>orig.txt</code> به دایرکتوری <code>otherdir/otherdir</code> . کپی با نام <code>new.txt</code> خواهد بود.	<code>cp orig.txt /otherdir/otherdir/new.txt</code>

نکته‌ی حیاتی برای درک این است که چگونه نام فایل مقصد مشخص می‌شود. این ممکن است در برخی موارد کمتر واضح باشد، زیرا مشخصات فایل و دایرکتوری ممکن است شبیه به هم باشند. به عنوان مثال، دستور زیر را در نظر بگیرید:

```
$ cp outline.pdf ~/publication
```

این دستور ممکن است به سه نتیجه‌ای که زیراً نشان داده شده است، منجر شود:

- اگر `~/publication` یک دایرکتوری باشد، نتیجه یک فایل به نام `~/publication/outline.pdf` خواهد بود.

نکته: اگر یک نام دایرکتوری را با یک کش (/) دنبال کنید، مانند `~/publication`، اگر `~/` وجود نداشته باشد یا یک فایل معمولی باشد، `cp` یک پیام خطاب برگرداند.

- اگر `~/` یک فایل باشد، نتیجه آن این است که این فایل توسط محتوای فایل `outline.pdf` جایگزین می‌شود.

نتایج اگر تازه کار هستید، ممکن است سردرگم‌کننده باشد. ما شما را تشویق می‌کنیم که با ایجاد یک دایرکتوری آزمایشی با استفاده از `mkdir`` (که در ادامه توضیح داده خواهد شد در "Creating Directories")

ایجاد زیردایرکتوری‌ها در این دایرکتوری، و کپی کردن فایل‌ها به این درخت آزمایشی از همهٔ این روش‌های ارجاع به فایل‌ها آزمایش کنید. (در اینجا استفاده از `touch` برای ایجاد فایل‌های آزمایشی مفید است.)

دستور `cp` امکانات زیادی برای تغییر عملکرد خود فراهم می‌کند. برخی از آپشن‌های مفیدتر این دستور امکان تغییر عملکرد دستور به شکل‌های مفید را فراهم می‌کنند:

**Force Overwrite**: استفاده از گزینه `force` یا `--force` سیستم را مجبور می‌کند که بدون سوال هر فایل موجودی را بازنویسی کند.

**Use Interactive Mode**: گزینه `--interactive` یا `'-i` باعث می‌شود که `cp` قبل از بازنویسی هر فایل موجود از شما سوال کند.

**Preserve Ownership and Permissions**: به طور معمول، کاربری که دستور `cp` را استفاده می‌کند، مالک یک فایل کپی شده است و از مجوزهای پیش‌فرض این حساب کاربری استفاده می‌کند. گزینه `--preserve` یا `'-p` مالکیت و مجوزها را در صورت امکان حفظ می‌کند.

نکته: فصل 13، "Creating Users and Groups" و "Setting Ownership and Permissions" در مورد مجوزهای فایل توضیح می‌دهد.

**Perform a Recursive Copy**: اگر از گزینه `R` یا `--recursive` استفاده کنید و یک دایرکتوری را به عنوان منبع مشخص کنید، تمام دایرکتوری با زیردایرکتوری‌هایش کپی می‌شود. اگرچه `cp` همچنین یک کپی بازگشتی انجام می‌دهد، اما رفتار آن با فایل‌هایی به غیر از فایل‌ها و دایرکتوری‌ها مشخص نیست. بیشتر اجرای‌های `cp` از `R` به عنوان مترادف `R` استفاده می‌کنند، اما این رفتار تضمین شده نیست.

**Perform an Archive Copy**: گزینه `a` یا `archive` مشابه `R` است، اما همچنین مالکیت را حفظ می‌کند و لینک‌ها را به همان شکل کپی می‌کند. گزینه `R` فایل‌ها را کپی می‌کند که لینک‌های Symbolic به آن‌ها اشاره دارند به جای خود لینک‌های نمادین. (لینک‌ها در ادامه این فصل، در "Using Links" به تفصیل توضیح داده شده‌اند).

**Perform an Update Copy**: گزینه `u` یا `update` به `cp` می‌گوید که فقط فایل را کپی کند اگر فایل اصلی جدیدتر از هدف باشد یا اگر هدف وجود نداشته باشد.

نکته: این لیست از گزینه‌های `cp` ناقص است اما بیشتر گزینه‌های مفید را پوشش می‌دهد. برای اطلاعات بیشتر در مورد گزینه‌های اضافی `cp`، به صفحه `man cp` مراجعه کنید.

## Moving and Renaming Files

در یک محیط متنی، از همان دستور "mv" برای جابجایی و تغییر نام فایل‌ها و دایرکتوری‌ها استفاده می‌شود. کاربرد آن مشابه دستور "cp" است. به عنوان مثال، اگر بخواهید فایل pdf outline را به مسیر ~/publication منتقل کنید، می‌توانید این دستور را وارد کنید:

```
$ mv outline.pdf ~/publication
```

اگر نام یک فایل را با مقصد مشخص کنید، فایل با جابجایی تغییر نام می‌یابد. اگر یک نام فایل مشخص کنید و دایرکتوری مقصد با دایرکتوری مبدأ یکسان باشد، فایل تغییر نام می‌یابد اما جابجا نمی‌شود. به عبارت دیگر، تأثیرات "mv" بسیار شبیه به "cp" هستند، با این تفاوت که فایل جدید جایگزین فایل اصلی می‌شود به جای اینکه آن را تکمیل کند.

در پشت صحنه، دستور "mv" عملیات زیر را انجام می‌دهد:

- وقتی منبع و هدف در همان فایل سیستم هستند، "mv" ورودی‌های دایرکتوری را بازنویسی می‌کند بدون اینکه واقعاً داده‌های فایل را جابجا کند.

نکته: لینوکس از "mv" برای تغییر نام فایل‌ها استفاده می‌کند زیرا این دو عمل وقتی که دایرکتوری‌های منبع و مقصد یکسان باشند، یکسان هستند.

- هنگامی که یک فایل را از یک فایل سیستم به فایل سیستم دیگر منتقل می‌کنید، `mv` فایل را کپی کرده و سپس فایل اصلی را حذف می‌کند.

--mv` اکثر آپشن‌هایی که دستور `cp` دارد را پشتیبانی می‌کند. از لیست قبلی، گزینه‌های `preserve`--archive` و `recursive` معنی ندارند، اما سایر گزینه‌ها قابل استفاده هستند.

## Using Links

گاهی اوقات مفید است که بتوانید به یک فایل با چند نام ارجاع کنید. به جای ایجاد چند نسخه از فایل، می‌توانید چندین لینک به یک فایل ایجاد کنید. لینوکس از دو نوع لینک پشتیبانی می‌کند که هر دو با دستور `ln` ایجاد می‌شوند:

**Hard Link**: هارد لینک یک ورودی تکراری در فهرست یا دایرکتوری است. هر دو ورودی به همان فایل اشاره دارند. این لینک‌ها با اتصال ساختارهای داده سطح پایین فایل‌سیستم کار می‌کنند و تنها می‌توانند در یک فایل‌سیستم وجود داشته باشند. در یک سناریوی هارد لینک، هیچ یک از دو نام فایل اولویتی نسبت به دیگری ندارند؛ هر دو به طور مستقیم به ساختارها و داده‌های فایل متصل می‌شوند. برای ایجاد یک هارد لینک از دستور `ln originname linkname` استفاده کنید، که در آن `originname` نام اصلی و `linkname` نام لینک جدید است.

**Symbolic Link**: (به نام لینک نمادین یا لینک سمبولیک شناخته می‌شود) یک فایل است که به یک فایل دیگر با نام ارجاع می‌دهد. به عبارت دیگر، لینک نمادین یک فایل است که نام یک فایل دیگر را نگه می‌دارد، و هنگامی که به یک برنامه می‌گویید که از یک فایل لینک نمادین بخواند یا به آن بنویسد، لینوکس دسترسی را به فایل اصلی هدایت می‌کند. از آنجا که لینک‌های نمادین با ارجاع به نام فایل کار می‌کنند، می‌توانند از مرزهای فایل‌سیستم عبور کنند. برای ایجاد یک لینک نمادین از دستور `ln -s originname linkname` استفاده کنید.

نکته: لینک‌های سمبليک مشابه شورتکات‌ها در دسکتاپ ویندوز هستند.

می‌توانید لینک‌ها را در لیست‌های طولانی دایرکتوری (با استفاده از گزینه -a برای ls) به چند روش تشخیص دهید. مثال زیر این را نشان می‌دهد:

```
$ ln report.odt hardlink.odt  
  
$ ln -s report.odt softlink.odt  
  
$ ls -l  
  
total 192  
  
-rw-r--r-- 2 rich users 94720 Sep 10 11:53 hardlink.odt  
  
-rw-r--r-- 2 rich users 94720 Sep 10 11:53 report.odt  
  
lrwxrwxrwx 1 rich users 10 Sep 10 11:54 softlink.odt ->  
report.odt
```

این مثال با یک فایل به نام report.odt شروع شد. دو دستور اول، دو لینک ایجاد کردند: یک هارد لینک (hardlink.odt) و یک لینک سمبليک (softlink.odt). تایپ -a همه سه فایل را نشان می‌دهد. فایل اصلی و هارد لینک می‌توانند به عنوان لینک با مقادار 2 در ستون دوم خروجی -a شناخته شوند؛ این ستون تعداد ورودی‌های نام فایل را که به فایل اشاره دارند، نشان می‌دهد، بنابراین مقادار بالاتر از 1 نشان‌دهنده وجود یک هارد لینک است. لینک سمبليک با حرف ا (یک L کوچک، نه رقم 1) در اولین کاراکتر رشته مجوز فایل softlink.odt مشخص می‌شود (lrwxrwxrwx). علاوه بر این، مشخصات فایل لینک نمادین حاوی یک اشاره صریح به فایل متصل شده است.

هر دو نوع لینک برای ارجاع به فایل‌ها با چند نام یا در چندین دایرکتوری مفید هستند. به عنوان مثال، اگر نامهای بنویسید که به چندین گیرنده ارسال می‌شود، ممکن است بخواهید کپی‌ها را در دایرکتوری‌های اختصاصی به هر گیرنده ذخیره کنید. در چنین شرایطی، هر نوع لینک احتمالاً کار خواهد کرد، اما هر نوع نکات مهمی دارد. مهمترین این نکات این است که اگر از لینک‌های سمبليک استفاده کنید، حذف فایل اصلی باعث شود که فایل به صورت کامل در دسترس نباشد؛ لینک‌های سمبليک باقی می‌مانند اما به یک فایلی که وجود ندارد اشاره می‌کنند. به عبارت دیگر، اگر از هارد لینک‌ها استفاده کنید، باید تمام نسخه‌های فایل را حذف کنید تا خود فایل حذف شود. این به این دلیل است که هارد لینک‌ها ورودی‌های تکراری دایرکتوری هستند که به همان فایل اشاره می‌کنند، در حالی که لینک‌های سمبليک فایل‌های جداگانه‌ای هستند که با نام به فایل اصلی اشاره می‌کنند.

اگر یک فایل را با دسترسی از طریق لینک سمبليک یا هر نام هارد لینک شده تغییر دهید، باید مطمئن شوید که برنامه‌ای که استفاده می‌کنید تغییرات را در فایل اصلی انجام خواهد داد. برخی از برنامه‌ها یک نسخه پشتیبان از فایل اصلی ایجاد می‌کنند که شما می‌توانید از آن برای بازیابی فایل اصلی در صورتی که متوجه

شوید تغییرات شما اشتباه بوده‌اند، استفاده کنید. بیشتر ادیتورها این کار را به گونه‌ای انجام می‌دهند که پشتیبان یک فایل جدید است و تغییرات را در فایل اصلی نوشته و بر روی آن اعمال می‌کند. با این حال، برخی از برنامه‌ها نام فایل اصلی را تغییر می‌دهند و سپس یک فایل جدید با تغییرات ایجاد می‌کنند. اگر یک برنامه این کار را انجام دهد و شما از طریق یک لینک به فایل دسترسی پیدا کرده باشید، فایل متصل به تغییرات شما تأثیری نخواهد دید. در صورت شک، برنامه خود را تست کنید تا اطمینان حاصل کنید که همانطور که انتظار دارید عمل می‌کند.

اگر می‌خواهید یک لینک به یک فهرست ایجاد کنید، بدانید که به طور عموم می‌توانید این کار را فقط از طریق لینک‌های سمبليک انجام دهید. هارد لینک‌ها بین فهرست‌ها از نظر ساختارهای داده سطح پایین برخی اوقات خطرناک هستند، بنابراین ابزار `In` فقط به کاربر اصلی اجازه ایجاد چنین لینک‌هایی را می‌دهد. حتی در این صورت، اکثر فایل سیستم‌ها ایجاد هارد لینک‌ها بین فهرست‌ها را اجازه نمی‌دهند، بنابراین حتی در عمل روت (superuser) هم معمولاً نمی‌تواند آن‌ها را ایجاد کند. با این حال، هر کاربر می‌تواند لینک‌های سمبليک به یک فهرست ایجاد کند.

توزيع‌های لینوکس در مکان‌های مختلف از لینک‌ها (عمدتاً لینک‌های سمبليک) استفاده می‌کنند. به عنوان مثال، اسکریپت‌های راه‌اندازی سیستم اغلب از طریق لینک‌های سمبليک اشاره می‌شوند که در دایرکتوری‌های اختصاصی برای شرایط خاص راه‌اندازی، به نام "runlevels" (سطوح اجرا) قرار دارند. مدیریت سطوح اجرا خارج از دامنه این کتاب است.

## Deleting Files

دستور `rm` فایل‌ها را در یک شل متنی حذف می‌کند. همانطور که انتظار می‌رود، شما نام یک یا چند فایل را به این دستور منتقل می‌کنید:

نکته: نام دستور `rm` به معنای حذف (remove) است.

```
$ rm outline.pdf outline.txt
```

این مثال دو فایل، `outline.pdf` و `outline.txt` را حذف می‌کند. اگر می‌خواهید یک درخت کامل از دایرکتوری‌ها را حذف کنید، می‌توانید به `rm` گزینه `-R`-`r`- یا `-r`- را به همراه نام یک دایرکتوری بدهید.

```
$ rm -r oldstuff/
```

آپشن `-a`- باعث می‌شود که `rm` قبل از حذف هر فایل سوال کند. این یک اقدام ایمنی مفید است. می‌توانید از گزینه `(--force)`-`f`- استفاده کنید تا این تنظیم را نادیده بگیرید، اگر `a`- به عنوان تنظیم پیش‌فرض تنظیم شده باشد. چندین گزینه دیگر هم برای دستور `rm` وجود دارد؛ برای یادگیری درباره آن‌ها، به صفحه راهنمای `man rm` آن مراجعه کنید.

نکته: گاهی اوقات توزیع‌ها گزینه `a`- را به صورت پیش‌فرض برای کاربر `root` تنظیم می‌کنند، اما برای کاربران عادی `n`- است.

مهم است که درک کنید که دستور `rm` هیچ قابلیتی مشابه سطل زباله یک فایل منیجر ندارد. پس از حذف یک فایل با `rm`, آن فایل از بین رفته و شما نمی‌توانید آن را به جزء استفاده از ابزارهای سطح پایین فایل‌سیستم بازیابی کنید و این موضوعی است که خارج از دامنه این کتاب است. بنابراین، هنگام استفاده از `rm` باید مواظب باشید مخصوصاً هنگام استفاده از آن با گزینه `-r` یا به عنوان `!root`

## Using Wildcards

یک wildcard نماد یا مجموعه‌ای از نمادها است که به جای دیگر کاراکترها قرار می‌گیرد. شما می‌توانید از wildcards برای ارجاع به فایل‌ها استفاده کنید. (استفاده از wildcards گاهی هم به نام globbing شناخته می‌شود.) سه کلاس از wildcards رایج در لینوکس عبارتند از:

? : یک علامت سوال (?) به عنوان یک کاراکتر قرار می‌گیرد. به عنوان مثال، `b?k` با `balk`, `buck`, `b??k` یا هر نام فایل چهار کاراکتری دیگری که با `b` شروع شده و با `k` ختم شده، مطابقت دارد.

\* : یک ستاره (\*) با هر کاراکتر یا مجموعه‌ای از کاراکترها، از جمله قادر هیچ گونه کاراکتری، مطابقت دارد. به عنوان مثال، `b*` با `balk`, `b?k` و `b??k` مطابقت دارد. همچنین `b*k` با `bk`, `bbk` و `bkk` مطابقت دارد.

**Bracketed Values**: کاراکترهای داخل براکت ([ ]) به طور معمول با هر کاراکتر در داخل مجموعه مطابقت دارند. به عنوان مثال، `b[a]o` با `balk` و `b[lo]k` با `buck` مطابقت دارد اما با `b` مطابقت ندارد. همچنین می‌توانید یک محدوده از مقادیر را مشخص کنید؛ به عنوان مثال، `b[a-z]ck` با `b` و `b[a-z]ack` با `b` و سایر نام‌های فایل چهار حرفی این شکل که کاراکتر دوم آن یک حرف کوچک است، مطابقت دارد. این با `b?ck` متفاوت است زیرا لینوکس در نام‌های فایل با حساسیت به بزرگ و کوچک حروف برخورد می‌کند و چون ? به هر کاراکتر (نه فقط هر حرف) مطابقت دارد، `b3ck` یا `b[A-Z]ck` با `b` مطابقت ندارد، در حالی که `b?ck` با این دو نام فایل مطابقت دارد.

Wildcards در شل اجرا می‌شوند و به دستوری که فراخوانی می‌کنید منتقل می‌شوند. به عنوان مثال، اگر `ls` را تایپ کنید و این وايد کارد با سه فایل `balk`, `book` و `buck` مطابقت داشته باشد، نتیجه تقریباً همانند این است که `ls` `balk book buck` را تایپ کرده باشید.

نکته: توسعه وايد کاردها توسط Bash می‌تواند به نتایج غیرمنتظره و گاهی اوقات ناخواسته منجر شود. به عنوان مثال، فرض کنید که می‌خواهید دو فایل را، که از طریق یک وايد کارد مشخص شده‌اند، به یک دایرکتوری دیگر کپی کنید، اما فراموش می‌کنید که مسیر مقصد را بدھید. دستور `cp` این دستور را به عنوان یک درخواست برای کپی کردن اولین فایل روی دومین فایل تفسیر خواهد کرد.

## Understanding Case Sensitivity

فایل‌سیستم‌های اصلی لینوکس حساس به حروف بزرگ و کوچک هستند، به این معنا که نام‌های فایل که تنها در حروف بزرگ یا کوچک اختلاف دارند، فایل‌های متمایز هستند. به عنوان مثال، یک دایرکتوری می‌تواند

فایل‌هایی با نام‌های Afile.TXT و Afile.txt، Afile.txt را در خود داشته باشد و هرکدام از آن‌ها یک فایل متمایز است. این حساسیت به حروف بزرگ و کوچک همچنین به این معناست که اگر شما نام یک فایل را تایپ کنید، باید آن را با حروف صحیح تایپ کنید. اگر یک فایل Afile.txt نام دارد اما شما نام آن را به عنوان Afile.txt تایپ کنید، برنامه‌ای که استفاده می‌کنید به شما خواهد گفت که فایل وجود ندارد. این موضوع با ویندوز یا (معمولًا) در مک متفاوت است، جایی که نام‌های فایل که تنها در حروف بزرگ یا کوچک اختلاف دارند یکی محسوب می‌شوند. در این سیستم‌عامل‌ها نیز نمی‌توانید دو فایل که تنها در حروف کوچک یا بزرگ اختلاف دارند را در یک دایرکتوری مشترک داشته باشید و می‌توانید نام یک فایل را با هر نوع حروف که دلتان بخواهد مشخص کنید. همچنین ویندوز یک نام فایل کوتاه (هشت حرف با یک پسوند اختیاری سه حرفی) برای هر فایل با نام بلند ایجاد می‌کند تا به نرم‌افزارهای قدیمی که تنها با این نام‌ها کار می‌کنند، کمک کند. لینوکس اینگونه نام‌های جایگزین را ایجاد نمی‌کند.

**نکته:** فایل‌سیستم سلسله‌مراتبی پلاس اپل (+HFS) همچنان دارای نسخه‌های حساس به حروف بزرگ و کوچک و نسخه‌های بی‌تفاوت به حروف بزرگ و کوچک است. اپل به طور پیش‌فرض از حالت بی‌تفاوت به حروف بزرگ و کوچک استفاده می‌کند.

حساسیت به حالت حروف بیشتر از همه به عنوان یک ویژگی از فایل‌سیستم می‌آید و نه از سیستم عامل. بنابراین، اگر به یک فایل‌سیستم غیر لینوکسی (روی یک دیسک پرتابل، یک پارتیشن غیر لینوکسی در یک کامپیوتر دوال بوت، یا با استفاده از یک فایل‌سیستم شبکه) دسترسی پیدا کنید، ممکن است متوجه شوید که قوانین بی‌تفاوت به حالت حروف بزرگ و کوچک اعمال می‌شود. این امر به ویژه زمانی ممکن است که به حجم‌های FAT و NTFS دسترسی پیدا کنید که در کامپیوترهای ویندوز، هارد دیسک‌های خارجی و درایوهای فلاش USB رایج هستند. یک تابع اضافی در این قاعده این است که بسیاری از برنامه‌های لینوکس، مانند Bash، حتی در فایل‌سیستم‌های بی‌تفاوت به حالت حروف، حساسیت به حالت حروف را فرض می‌کنند. ویژگی‌هایی همچون تکمیل دستورات، که در فصل 5، "Getting to Know the Command Line" توضیح داده شده‌اند، ممکن است تنها زمانی کار کنند که شما از حالت حروف استفاده کنید که در آن فایل‌ها ذخیره شده‌اند، حتی در فایل‌سیستم‌های بی‌تفاوت به حالت حروف بزرگ و کوچک.

به طور عادی، حساسیت به حالت حروف مشکلات واقعی ایجاد نمی‌کند، به خصوص اگر از برنامه‌های گرافیکی استفاده کنید که به شما امکان انتخاب فایل با استفاده از کلیک دکمه فراهم می‌کنند. با این حال، شما باید از این مسائل اطلاع داشته باشید زمانی که فایل‌ها یا دایرکتوری‌ها را به FAT، NTFS، و +HFS یا دیگر فایل‌سیستم‌های بی‌تفاوت به حالت حروف می‌آورید. اگر یک دایرکتوری که می‌خواهید کپی کنید دارای فایل‌هایی با نام‌هایی که تنها در حالت حروف متفاوت هستند، شما خود را با یک دیسک که تنها یکی از فایل‌ها را شامل می‌شود خواهید یافت.

## Manipulating Directories

احتمالاً با مفهوم دایرکتوری‌ها آشنا هستید، گرچه ممکن است به عنوان "فولدرها" به آن‌ها فکر کنید، زیرا بیشتر فایل منیجرهای گرافیکی از آیکون‌های پوشش‌های فایل برای نمایش دایرکتوری‌ها استفاده می‌کنند. طبیعتاً، لینوکس دستورات متعددی را برای کنترل دایرکتوری‌ها فراهم کرده است. این شامل دستورات خاص

دایرکتوری برای ایجاد و حذف دایرکتوری‌ها است، همچنین استفاده از برخی از دستورات مدیریت فایل که قبل از شرح داده شده‌اند برای مدیریت دایرکتوری‌ها می‌شود.

## Creating Directories

می‌توانید از دستور `mkdir` برای ایجاد یک دایرکتوری استفاده کنید. به طور عادی، از این دستور با تایپ نام یک یا چند دایرکتوری پس از دستور استفاده خواهید کرد:

```
$ mkdir newdir
```

```
$ mkdir dirone newdir/dirtwo
```

در این دستور نمونه اول، تنها یک دایرکتوری جدید به نام `newdir` ایجاد می‌شود که سپس در دایرکتوری فعلی قرار می‌گیرد. در دستور نمونه دوم، دو دایرکتوری جدید ایجاد می‌شود: `newdir/dirtwo` و `dirone`. در این مثال، `mkdir` دایرکتوری جدید به نام `dirtwo` را درون دایرکتوری `newdir` ایجاد می‌کند که با دستور قبلی ایجاد شده است.

نکته: فصل 5 حاوی اطلاعاتی در مورد نحوه مشخص کردن موقعیت‌ها به غیر از دایرکتوری فعلی است، همچنین نحوه تغییر دایرکتوری فعلی با دستور `cd` توضیح داده شده است.

در اکثر موارد، از `mkdir` بدون گزینه‌ها به جز نام یک دایرکتوری استفاده خواهید کرد، اما می‌توانید رفتار آن را به چند روش تغییر دهید:

**Set Mode**: گزینه `-m mode` یا `-mode=mode` موجب ایجاد دایرکتوری با مجوز مشخص شده می‌شود که به صورت یک عدد اوکتال (Octal) بیان شده است. (فصل 14 این موضوعات را به طور دقیق‌تر توضیح می‌دهد.)

**Create Parent Directories**: به طور معمول، اگر شما ایجاد یک دایرکتوری درون یک دایرکتوری که وجود ندارد را مشخص کنید، `mkdir` با یک پیغام خطاب به اسم `No such file or directory` می‌دهد و دایرکتوری را ایجاد نمی‌کند. اگر اما گزینه `-p` یا `-parents` را اضافه کنید، `mkdir` دایرکتوری `parent` را ایجاد می‌کند. به عنوان مثال، تایپ کردن `mkdir first/second` اگر `first` وجود نداشته باشد با یک پیغام خطاب همراه می‌شود، اما تایپ کردن `mkdir -p first/second` موفقیت‌آمیز است و هر دو `first` و زیردایرکتوری `second` را ایجاد می‌کند.

## Deleting Directories

دستور `rmdir` مخالف دستور `mkdir` است؛ دایرکتوری را حذف می‌کند. برای استفاده از آن، معمولاً دستور را تایپ می‌کنید و سپس اسمی یک یا چند دایرکتوری که می‌خواهید حذف کنید را وارد می‌کنید:

```
$ rmdir dirone
```

```
$ rmdir newdir/dirtwo newdir
```

این نمونه‌ها سه دایرکتوری را که توسط دستورهای `mkdir` نشان داده شده‌اند، حذف می‌کنند.

مانند `mkdir`، `rmdir` تعداد اندکی از گزینه‌ها را پشتیبانی می‌کند، اهمیت بیشترین آنها مربوط به انجام وظایف زیر است:

نادیده گرفتن خطاهای دایرکتوری‌های غیر خالی: به طور عادی، اگر یک دایرکتوری فایل یا دایرکتوری‌های دیگری را شامل شود، `rmdir` آن را حذف نمی‌کند و یک پیام خطا برمنی‌گرداند. با گزینه `-ignore-fail-on-non-empty`، همچنان دایرکتوری `rmdir` حذف نمی‌شود، اما پیام خطا برمنی‌گرداند.

حذف درخت: گزینه `-p` یا `-parents` باعث می‌شود که `rmdir` یک درخت دایرکتوری را حذف کند. به عنوان مثال، تایپ کردن `rmdir -p newdir/dirtwo` باعث می‌شود `rmdir` درخت `newdir/dirtwo` را حذف کند، سپس `newdir` را. می‌توانید از این دستور به جای دستور دومی که در ابتداء نشان داده شد، برای حذف هر دوی این دایرکتوری‌ها استفاده کنید.

باید درک کنید که `rmdir` تنها می‌تواند دایرکتوری‌های خالی را حذف کند؛ اگر یک دایرکتوری حاوی هر گونه فایلی باشد، کار نخواهد کرد. (با این حال، می‌توانید از گزینه `-r` استفاده کنید تا یک مجموعه از دایرکتوری‌های تو در تو را حذف کنید، تا زمانی که هیچ‌کدام از آن‌ها فایلی غیر از دایرکتوری نداشته باشند). البته در واقعیت، احتمالاً می‌خواهید درختان دایرکتوری که حاوی فایل هستند را حذف کنید. در چنین مواردی، می‌توانید از دستور `rm -r` که در ابتداء در "Deleting Files" توضیح داده شد، به همراه گزینه `-r` (یا `-R`) استفاده کنید:

```
$ rm -r newdir
```

این دستور `newdir` و هر فایل یا زیردایرکتوری‌ای که ممکن است شامل شود را حذف می‌کند. این موضوع باعث می‌شود دستور `rm -r` یا گزینه `-r` آن به طور پتانسیل خطرناک عمل کند، بنابراین باید ویژه‌ترین احتیاط را در هنگام استفاده از آن داشته باشید.

## Real World Scenario

### Linux Security Features

وقتی به عنوان یک کاربر عادی وارد سیستم می‌شوید، ممکن است با استفاده اشتباه از دستور `rm` یا دستورهای مختلف دیگر، فایل‌های خود را حذف کنید. با این حال، می‌توانید به سیستم عامل لینوکس خود آسیب جدی وارد کنید. این به این دلیل است که یونیکس به عنوان یک سیستم عامل چندکاربره با توجه به ویژگی‌های امنیتی چندکاربره طراحی شده بود و چون لینوکس یک کلون از یونیکس است، این ویژگی‌های امنیتی را به ارت برده است. در میان این ویژگی‌ها مفهوم مالکیت فایل و دسترسی‌های فایل وجود دارد. شما تنها می‌توانید فایل‌های خودتان را حذف کنید - یا به عبارت دقیق‌تر، تنها در صورتی می‌توانید فایل‌ها را حذف کنید که به نوشتن در دایرکتوری‌هایی که در آنها قرار دارند دسترسی پیدا کنید. شما به این مجوز در دایرکتوری خانه‌ی خود دسترسی دارید اما به دایرکتوری‌هایی که فایل‌های سیستم

لینوکس در آنها قرار دارند دسترسی ندارید. بنابراین، می‌توانید به این فایل‌های سیستم لینوکس آسیب بزنید.

فصل 13 به این مفاهیم به صورت دقیق تر پرداخته است. فصل 13 همچنین توضیح می‌دهد چگونه می‌توانید قدرت مدیریت کامپیوتر را به دست آورید. با این قدرت، قابلیت آسیب زدن به سیستم نیز همراه است، بنابراین باید دقت کنید و تنها در صورت لزوم اقدام به آن کنید.

## Managing Directories

دایرکتوری‌ها فقط فایل‌های ویژه‌ای هستند که دیگر فایل‌ها را در خود نگه می‌دارند. این بدان معناست که می‌توانید از اکثر ابزارهای مدیریت فایل که در سایر قسمت‌های این فصل شرح داده شده‌اند، برای مدیریت دایرکتوری‌ها استفاده کنید. با این حال، توجه به نکات زیر حائز اهمیت است:

- شما می‌توانید از دستور touch برای بهروزرسانی برچسب‌های زمانی یک فهرست استفاده کنید، اما می‌توانید از touch برای ایجاد یک دایرکتوری استفاده کنید؛ این وظیفه توسط mkdir انجام می‌شود.
- شما می‌توانید از cp برای کپی کردن یک دایرکتوری استفاده کنید؛ با این حال، باید از گزینه‌های -R، -a، -archive یا -recursive استفاده کنید تا فهرست و تمام محتویات آن را کپی کنید.
- شما می‌توانید از mv برای جابجایی یا تغییر نام یک دایرکتوری استفاده کنید.
- شما می‌توانید از ln با گزینه -s برای ایجاد یک لینک سمبولیک به یک دایرکتوری استفاده کنید. هر چند که هیچ یک از فایل‌سیستم‌های معمول لینوکس، پشتیبانی از هارد لینک‌ها به دایرکتوری‌ها را ندارد.

به عنوان یک مثال، فرض کنید که یک فهرست در دایرکتوری خانه‌تان به نام Music/Satchmo دارید که شامل فایل‌های موسیقی لوئی آرمستانگ است. شما می‌خواهید این دایرکتوری را مرتب کنید تا فایل‌ها زیر نام خانوادگی اجرا کننده ظاهر شوند، اما می‌خواهید به فایل‌ها با نام Satchmo دسترسی داشته باشید، زیرا پخش‌کننده‌های موسیقی به این شکل به آن‌ها ارجاع داده‌اند. می‌توانید دستورات زیر را تایپ کنید تا این هدف را دستیابی کنید:

```
$ cd ~/Music  
$ mv Satchmo Armstrong  
$ ln -s Armstrong Satchmo
```

به عنوان یک گزینه دیگر، می‌توانید دستور اول را حذف کنید و مسیر کامل به هر یک از دایرکتوری‌ها یا لینک‌ها را در دستورهای mv و ln مشخص کنید. در حالت فعلی، دو دستور اول این دستورات دایرکتوری

را به ~/~/Music/Armstrong تغییر نام می‌دهند. دستورنهایی یک لینک سمبولیک، ایجاد می‌کند که به ~/~/Music/Armstrong اشاره دارد.

# CHAPTER 8

## Searching, Extracting, and Archiving Data

یک بخش مهم از کار هر سیستم عامل، از جمله لینوکس، ذخیره، مدیریت و تجزیه و تحلیل دادهها است. داده به دلیل دانشی که می‌توان از آن استخراج کرد، ارزشمند است؛ بنابراین شما باید بتوانید به درستی در داده‌ها جستجو کرده و آنها را استخراج و حفاظت کنید. این فصل به بررسی برخی از ابزارهایی می‌پردازد که می‌توانید از آنها برای جستجو، استخراج و آرشیو کردن داده‌ها استفاده کنید.

این فصل با یک نگاه به عبارات باقاعدۀ شروع می‌شود که یک راه برای توصیف الگوهایی است که ممکن است در فایل‌های داده بخواهید بگردید. شما می‌توانید از عبارات باقاعدۀ با بسیاری از دستورات استفاده کنید، از جمله دو دستور `find` و `grep` که به تفصیل توضیح داده شده‌اند. این فصل همچنین ابزارهایی را معرفی می‌کند که می‌توانید از آنها برای تغییر جهت ورودی و خروجی برنامه‌ها استفاده کنید، که در بسیاری از موقع حیاتی است. در نهایت، ابزارهایی برای ایجاد فایل‌های آرشیو شرح داده شده‌اند که می‌توانند در انتقال بسیاری از فایل‌ها در شبکه یا ایجاد نسخه‌های پشتیبان مفید باشد.

### Using Regular Expressions

برخی از برنامه‌های لینوکس از عبارات باقاعدۀ استفاده می‌کنند که ابزارهایی برای بیان الگوها در متن هستند. عبارات باقاعدۀ از لحاظ اصولی شبیه به وایلدکاردها هستند که می‌توانند برای مشخص کردن چندین نام فایل استفاده شوند، همانطور که در فصل 7، "Managing Files" توضیح داده شده است. در ساده‌ترین حالت، عبارات باقاعدۀ می‌توانند متن ساده باشند بدون نمادهای اضافی، با این حال برخی از کarakترها برای نمایش الگوها استفاده می‌شوند.

نکته: گاهی اوقات در مستندات از اختصار "regexp" برای اشاره به عبارت باقاعدۀ استفاده می‌شود.

دو نوع عبارت باقاعدۀ رایج هستند: ابتدایی و گستردۀ. نوعی که باید استفاده کنید، به برنامه بستگی دارد. برخی فقط یک فرم عبارت را قبول می‌کنند، در حالی که دیگران می‌توانند از هر دو نوع استفاده کنند (بسته به گزینه‌های ارسال شده به برنامه). تفاوت‌ها بین فرم‌های ابتدایی و گستردۀ ممکن است پیچیده و ظریف باشد، اما اصول اساسی هر دو مشابه هستند.

ساده‌ترین نوع عبارت باقاعدۀ یک رشته حروف الفباوی یا عددی است، مانند "HWaddr" یا "Linux3". این عبارات باقاعدۀ با هر رشته‌ای با اندازه مشابه یا بزرگتر که حاوی عبارت باقاعدۀ باشد، مطابقت دارند. به عنوان مثال، عبارت "The HWaddr is unknown" با "HWaddr" و "This is the HWaddr" مطابقت دارد. قدرت واقعی عبارات باقاعدۀ در استفاده از حروف غیر الفباوی است که قوانین تطابق پیشرفته را فعال می‌کنند.

قابلیت‌های قدرتمندترین عبارات منظم اصلی عبارات باقاعده عبارات زیر را شامل می‌شود:

**Bracket Expressions**: کاراکترهای مشخص شده درون براکت ([ ]), به نام یک عبارت باقاعده را تشکیل می‌دهند که با هر یک از کاراکترهای درون براکت مطابقت دارد. به عنوان مثال، عبارت باقاعده  $b[aeiou]g$  با کلمات  $beg$ ,  $big$ ,  $bog$  و  $bug$  مطابقت دارد. براکت‌ها نمایانگر یک کاراکتر درون کلمه هستند. بنابراین، عبارت باقاعده  $p[ai]n$  با کلمات  $pan$  و  $pin$  مطابقت دارد اما با  $paint$  مطابقت ندارد. اضافه کردن نشان **Caret** باقاعده  $p[ai]^n$  با کلمات  $pan$  و  $pin$  مطابقت دارد اما با  $paint$  مطابقت ندارد. پس از براکت ابتدایی، با هر کاراکتر به جز کاراکترهای مشخص شده مطابقت می‌یابد. به عنوان مثال،  $(^)$  پس از براکت ابتدایی، با هر کاراکتر به جز کاراکترهای مشخص شده مطابقت می‌یابد. به عنوان مثال،  $b[aeiou]g$  با  $bAg$  یا  $bbg$  یا  $b[^aeiou]g$  با  $beg$  اما با  $bag$  مطابقت ندارد.

**Range Expressions**: یک نوع از عبارت باقاعده "Bracket Expressions" است. به جای فهرست کردن هر کاراکتری که با آن مطابقت دارد، عبارتهای محدوده شروع و پایان را با یک خط تیره (-) جدا کرده و مشخص می‌کنند، مانند  $a[2-4]z$ . این عبارت باقاعده با  $a3z$ ,  $a2z$ ,  $aQz$  و  $a.z$  مطابقت دارد.

**Any Single Character**: نقطه (. ) هر کاراکتر تنها را به جز یک کاراکتر جدید مطابقت می‌دهد. به عنوان مثال،  $a.a2z$ ,  $abz$  یا هر رشته سه کاراکتری دیگر که با  $a$  شروع شده و با  $z$  ختم می‌شود، مطابقت دارد.

نکته: newline یک کاراکتر ویژه پنهان است که در فایل‌های متنی استفاده می‌شود. هنگامی که یک فایل متنی نمایش داده می‌شود، معمولاً یک کاراکتر newline در انتهای هر خط است که اغلب باعث نمایش هر خط زیر خط قبلی می‌شود.

**شروع و پایان خط**: یک خط متن (گاهی اوقات به آن رکورد هم گفته می‌شود) شامل تمام کاراکترها است که قبل از پایان با یک کاراکتر newline ختم می‌شود. زمانی که درون براکت نیست استفاده می‌شود، علامت  $(^)$  نمایانگر شروع یک خط است. علامت دلار (\$) نیز پایان یک خط را نشان می‌دهد. به عنوان مثال،  $bag^$  با  $bag$  فقط در صورتی مطابقت دارد که در ابتدای یک خط از کاراکترها باشد، در حالی که  $$bag$  فقط در صورتی مطابقت دارد که در انتهای یک خط از کاراکترها باشد.

تکرار: یک عبارت باقاعده کامل یا جزئی ممکن است با یک نماد ویژه دنبال شود تا تکرار مورد تطابق نشان داده شود. مخصوصاً یک استریک (\*) صفر یا چند تطابق را نشان می‌دهد. اغلب استریک با نقطه (مانند \*.\*) ترکیب می‌شود تا تطابق با هر زیررشته‌ای مشخص شود. به عنوان مثال،  $A.*Lincoln$  با هر رشته‌ای که شامل  $A$  و  $Lincoln$  به همین ترتیب باشد تطابق دارد - آب لینکلن و آبراهام لینکلن تنها دو تطابق ممکن هستند.

**Escaping**: در صورتی که می‌خواهید یکی از کاراکترهای ویژه را مطابقت دهید، مانند یک نقطه، شما باید از اسکیپ (escape) استفاده کنید؛ به عبارت دیگر، باید آن را با یک بک اسلش (\) پیش از آن قرار دهید. به عنوان مثال، برای مطابقت با نام هاست کامپیوتر (مثل [www.sybex.com](http://www.sybex.com))، شما باید همانند [www\sybex\com](http://www\sybex\com) انجام دهید.

عبارت‌های باقاعده گسترده (extended regular expressions) ویژگی‌های بیشتری اضافه می‌کنند که می‌توانید از آنها برای مطابقت با شبیوهای دیگر استفاده کنید:

اپراتورهای تکرار اضافی: اپراتورهای تکرار دیگر مانند یک علامت ستاره (\*) عمل می‌کنند، اما فقط با تعداد خاصی از مطابقتها متناسب هستند. به طور خاص، یک علامت پلاس (+) با یک یا چند تطابق متناظر می‌شود، و یک علامت سوال (?) تعداد تطابق‌ها را صفر یا یک مشخص می‌کند.

**رشته‌های ممکن چندگانه:** خط عمودی (|) دو تطابق ممکن را جدا می‌کند؛ به عنوان مثال، car|truck همزمان با truck یا car مطابق می‌شود.

**پرانترها:** پرانترهای معمولی (( )) زیرعبارت‌های فرعی را در برمی‌گیرند. پرانترها اغلب برای مشخص کردن نحوه اعمال اپراتورها استفاده می‌شوند؛ به عنوان مثال، می‌توانید پرانترها را در اطراف یک گروه از کلماتی که با خط عمودی ادغام شده‌اند بگذارید تا اطمینان حاصل شود که کلمات به عنوان یک گروه مورد استفاده قرار گرفته‌اند و هرکدام ممکن است مطابقت داشته باشند، بدون درگیر کردن بخش‌های اطراف عبارت باقاعده.

**نکته:** اگر از یک عبارت باقاعده گستردۀ با دستور grep استفاده کنید، باید گزینه E- را نیز اضافه کنید.

استفاده از عبارات باقاعدۀ ساده یا گستردۀ بستگی به این دارد که برنامه از کدام یک پشتیبانی می‌کند. با برنامه‌هایی مانند grep که هر دو را پشتیبانی می‌کنند، می‌توانید از هرکدام استفاده کنید؛ انتخاب شما به‌طور عمده از نظر ترجیح شخصی است. توجه داشته باشید که یک عبارت باقاعدۀ که شامل کاراکترهای مرتبط با عبارات باقاعدۀ گستردۀ باشد، بسته به اینکه از کدام نوع استفاده می‌شود، به طرز متفاوتی تفسیر خواهد شد. بنابراین، مهم است که بدانید برنامه از کدام نوع پشتیبانی می‌کند یا چگونه می‌توانید انتخاب کنید که از کدام نوع استفاده شود اگر برنامه از هر دو نوع پشتیبانی کند.

قوانین عبارات باقاعدۀ ممکن است گیج‌کننده باشند، بهویژه زمانی که برای اولین بار با آنها آشنا می‌شویم. برخی از نمونه‌های استفاده از آنها، در زمینه برنامه‌هایی که از آنها استفاده می‌شود، به شما کمک می‌کند. بخش بعدی این نمونه‌ها را با ارجاع به برنامه grep ارائه می‌دهد.

## Searching For and Extracting Data

دستور grep از عبارات باقاعدۀ استفاده می‌کند و در یافتن داده‌ها کمک می‌کند. این ابزار فایل‌ها را با اسکن محتوای آنها پیدا می‌کند. همچنین برنامه grep برخی از داده‌های موجود در فایل‌ها را نیز بازمی‌گرداند، که اگر می‌خواهید فقط یک کمی اطلاعات را از یک فایل یا خروجی یک برنامه استخراج کنید، ممکن است مفید باشد.

همانطور که از نامش پیداست، دستور find فایل‌ها را پیدا می‌کند و از ویژگی‌های سطحی مانند نام فایل و نشانگرهای تاریخ فایل استفاده می‌کند. دستور wc آمارهای ابتدایی کلمات در فایل‌های متنی را ارائه می‌دهد. برای استخراج آیتم‌های داده‌ها از خطوط یک فایل، دستور cut مفید است. دو دستور اضافی، sort و cat، امکان مدیریت نمایش داده‌های حاصل را فراهم می‌کنند، که می‌توانند در جستجوی شما مفید باشد.

**نکته:** برخلاف grep، دستور find به طور پیش‌فرض از عبارات باقاعدۀ استفاده نمی‌کند. با این حال، این دستور با استفاده از یک مکانیزم مشابه، پشتیبانی از الگوها را فراهم می‌کند.

## Using grep

دستور grep برای جستجو در فایل‌ها به دنبال یک رشته مشخص می‌گردد و نام فایل و (اگر فایل متنی باشد) خط حاوی آن رشته را باز می‌گرداند. همچنین می‌توانید از grep برای جستجوی یک رشته مشخص در یک فایل خاص استفاده کنید. برای استفاده از grep، نام دستور، یک مجموعه از اختیارات سوئیچ‌ها (آپشن‌ها)، یک عبارت باقاعده و یک مشخصه نام فایل اختیاری تایپ می‌کنید. دستور grep از تعداد زیادی از اختیارات پشتیبانی می‌کند، از جمله مهم‌ترین آنها در جدول 8.1 آمده است.

Table 8.1 Common grep options

Option (long form)	Option (short form)	Description
--count	-c	تعداد خطوطی که مطابقت دارند را به جای خطوطی که حاوی مطابقت با عبارت باقاعده هستند نشان می‌دهد.
--file=file	-f file	الگوی ورودی را از فایل مشخص شده برای الگو به جای کامند لاین می‌گیرد. دستور fgrep یک میانبر برای این گزینه است.
--ignorecase	-i	یک جستجوی غیر حساس به بزرگی و کوچکی حروف انجام می‌دهد.
--recursive	-R or -r	به جای اینکه فقط در دایرکتوری مشخص جستجو کند، در دایرکتوری مشخص و تمام زیردایرکتوری‌ها جستجو می‌کند.
--extended-regexp	-E	به منظور استفاده از یک عبارت باقاعده گسترده، این گزینه را ارسال کنید. به عنوان جایگزین، می‌توانید به جای `grep` از `egrep` استفاده کنید؛ این دستور نسخه‌ای تغییر یافته است که به صورت پیش‌فرض از عبارت‌های باقاعده گسترده استفاده می‌کند.

نکته: اگر شما نام یک فایل را مشخص نکنید، `grep` از ورودی استاندارد استفاده می‌کند. این می‌تواند با کاربرد داشته باشد، که در "Redirecting Input and Output" به زودی توضیح داده خواهد شد.

یک مثال ساده از `grep` با یک عبارت باقاعده بدون مؤلفه‌های ویژه به صورت زیر است:

```
$ grep -r bash /etc/
```

این مثال تمام فایل‌های موجود در `/etc/` که شامل رشته "bash" هستند را پیدا می‌کند. چون این مثال شامل گزینه `-r` است، به صورت بازگشتی جستجو می‌کند، بنابراین `grep` در فایل‌های زیرشاخه‌های `/etc/` و همچنین فایل‌های خود `/etc/` جستجو می‌کند. برای هر فایل متنی که شامل رشته باشد، خطی که حاوی این رشته است چاپ می‌شود.

نکته: کاربران عادی قادر به خواندن برخی از فایل‌ها در `/etc/` نیستند. اگر این دستور را به عنوان یک کاربر عادی وارد کنید، پیام‌های خط مرتبه با عدم توانایی `grep` در باز کردن برخی از فایل‌ها را خواهید دید.

فرض کنید می‌خواهید تمام فایل‌های موجود در `/etc/` را که حاوی رشته `bash` یا `dash` هستند، پیدا کنید. می‌توانید دستور زیر را وارد کنید که از یک عبارت باقاعده با استفاده از براکت برای مشخص کردن هر دو دستگاه مختلف استفاده می‌کند:

```
$ grep -r [bd]ash /etc/
```

یک مثال پیچیده‌تر جستجوی خطوطی است که فایل `/etc/passwd` حاوی کلمات `mail` یا `games` و، بعد از آن در همان خط، کلمه `nologin` را دارند. این کار نیازمند نوشтар گسترده عبارت باقاعده است؛ دستور به صورت زیر است:

```
$ grep -E "(games|mail).*nologin" /etc/passwd
```

نکته: اگر این دستور را در کامپیوتر خود وارد کنید، ممکن است به دلیل پیکربندی توزیع شما هیچ تطابقی پیدا نکند. به جای کلمات `mail` و `games`، کلمات دیگری را درون پرانتز امتحان کنید، مانند `nobody` یا `lp`.

دستور فوق ویژگی دیگری را نشان می‌دهد که ممکن است نیاز به استفاده از آن داشته باشد: نقل قول در شل (Shell Quoting). زیرا کامند لاین از برخی از نمادها، مانند خط عمودی `(`) و ستاره `(\*)`، برای اهداف خود استفاده می‌کند، باید برخی از عبارات باقاعده را در داخل نقل قول‌ها قرار دهیم. در غیر این صورت، کامند لاین عبارت باقاعده را به عنوان دستورات کامند لاین می‌پذیرد. نقل قول در کامند لاین برای برنامه‌های دیگری نیز مفید است که ممکن است از نمادهایی با معنای ویژه برای کامند لاین استفاده کنند، مانند دستور `echo` (که در فصل 11، "Creating Scripts" بحث شده است).

می‌توانید از دستور `grep` به همراه دستوراتی که حجم زیادی از خروجی را تولید می‌کنند استفاده کنید تا از آن خروجی برای یافتن مواد مهم برای شما استفاده کنید. (چندین مثال در سراسر این کتاب از این تکنیک

استفاده می‌کنند). برای انجام این کار، شما باید از هدایت ورودی و خروجی استفاده کنید. این موضوع (همراه با مثال‌های اضافی grep) در بخش بعدی "Redirecting Input and Output" بررسی می‌شود.

## Using Find

ابزار find روشی بدون درنگ برای پیدا کردن فایل‌ها اجرا می‌کند. این برنامه با جستجو در درخت دایرکتوری مشخص شده، بررسی نام فایل‌ها، تاریخ ایجاد فایل و غیره، به دنبال فایل‌هایی می‌گردد که با معیارهای مشخص شده مطابقت داشته باشند. به دلیل این روش عملکرد، find تمایل دارد که کند باشد. با این حال، این ابزار انعطاف‌پذیر است و احتمالاً موفق خواهد شد، با فرض اینکه فایل مورد نظر وجود داشته باشد. برای استفاده از find، نام آن را تایپ کنید، اختیاراً به همراه یک مسیر درخت دایرکتوری (گاهی اوقات به عنوان starting point نامیده می‌شود) و یک سری گزینه، که برخی از آن‌ها از مشخصاتی استفاده می‌کنند که شبیه به عبارات باقاعده است.

نکته: در عمل، شما باید از یک مسیر درخت دایرکتوری یا یک معیار جستجو و اغلب هر دو را با استفاده از find استفاده کنید.

شما می‌توانید یک یا چند مسیر را مشخص کنید که find باید در آن‌ها عمل کند؛ برنامه عملیات خود را به این مسیرها محدود می‌کند. صفحه راهنمای find شامل اطلاعاتی درباره معیارهای جستجو است، اما جدول 8.2 خلاصه‌ای از معیارهای متداول را ارائه می‌دهد.

Table 8.2 Common find search criteria

Option	Description
-name pattern	جستجو برای پیدا کردن فایل‌ها با استفاده از نام‌هایشان. این کار با پیدا کردن فایل‌هایی که الگوی مشخص شده را مطابقت دارند، انجام می‌شود. این الگو یک الگوی ویژه خودکار شل است، و نه یک عبارت باقاعده.
-perm mode	برای پیدا کردن فایل‌هایی که دارای مجوزهای خاصی هستند، از عبارت -perm mode استفاده کنید. این حالت می‌تواند به شکل نمادین یا اکتال ارائه شود. اگر حالت را با + شروع کنید، find فایل‌هایی را پیدا می‌کند که هر بیت مجوز مشخص شده تنظیم شده است. اگر حالت را با - شروع کنید، find فایل‌هایی را پیدا می‌کند که تمام بیت‌های مجوز مشخص شده تنظیم شده‌اند. (فصل 14، "Setting Ownership and Permissions"، مجوزهای فایل را پوشش می‌دهد).
-size n	برای جستجوی فایل‌ها بر اساس اندازه، به طور معمول n به صورت بلوك‌های 512 باشی می‌شود، اما شما می‌توانید این را با پیگیری مقدار با یک کد حروفی مانند c برای کاراکترها (بايت‌ها) یا k برای کیلوبایت‌ها تغییر دهید.
-group name	برای جستجوی فایل‌هایی که به گروه مشخصی تعلق دارند، از گزینه `group` استفاده کنید.

-gid GID	برای جستجوی فایل‌هایی که شناسه گروه (GID) آنها به GID تنظیم شده است، از گزینه `gid` استفاده کنید.
-user name	برای جستجوی فایل‌هایی که متعلق به کاربر مشخصی هستند، از گزینه `user` استفاده کنید.
-uid UID	برای جستجوی فایل‌هایی که شناسه کاربری (UID) آنها برابر با شماره مورد نظر است، از گزینه `uid` استفاده کنید.
-maxdepth levels	برای محدود کردن جستجو در یک دایرکتوری و شاید تعداد محدودی زیردایرکتوری، می‌توانید از گزینه `maxdepth` استفاده کنید.

خیلی گزینه‌ها و تنظیمات دیگر وجود دارد؛ `find` یک دستور قدرتمند است. به عنوان مثال برای استفاده از آن، در نظر بگیرید که می‌خواهید تمام فایل‌های اسکریپت پایتون که به طور معمول دارای نام‌هایی با پسوند `.py` هستند، در تمام دایرکتوری‌های خانه کاربران را پیدا کنید. اگر این دایرکتوری‌های خانه در دایرکتوری `/home` قرار داشته باشند، می‌توانید دستور زیر را وارد کنید:

```
# find /home -name "*.py"
```

نتیجه این دستور یک لیست از تمام فایل‌هایی خواهد بود که نام آن‌ها با `.py` پایان می‌یابد و در دایرکتوری `/home` قرار دارند. توجه کنید که با استفاده از نقل قول در کامند لاین از `"*.py"` استفاده شده است □ زیرا علامت ستاره (\*) معنای خاصی برای کامند لاین دارد.

نکته: اگر شما اجازه‌ی لیست کردن محتوای یک دایرکتوری را نداشته باشید، `find` نام آن دایرکتوری و پیام `Permission denied` را برمی‌گرداند.

## Using wc

اندازه‌ی یک فایل به بایت، که توسط `ls` نمایش داده می‌شود و یا با استفاده از `find` جستجو می‌شود، معیار مفیدی است. اما این مقدار اندازه برای فایل‌های متنی همیشه مفیدترین معیار نیست. ممکن است نیاز داشته باشید که بدانید چند کلمه یا خط در یک فایل متنی وجود دارد، مثلًا برای اینکه بدانید یک داکیومنت متنی چند صفحه را اشغال خواهد کرد وقتی که با سرعت چاپ 52 خط در هر صفحه چاپ شود. این اطلاعات توسط ابزار `WC` فراهم می‌شود. به عنوان مثال، برای کشف اطلاعات فایل جدیدی با نام `newfile.txt` در دایرکتوری کنونی خود:

```
$ wc newfile.txt
```

```
37 59 1990 newfile.txt
```

این خروجی نشان می‌دهد که فایل `newfile.txt` شامل 37 خط، 59 کلمه و 1990 بایت است. به طور پیش‌فرض، `WC` تعداد خطوط، کلمات و بایت‌ها را برای هر فایلی که به آن ارسال می‌شود نمایش می‌دهد.

شما می‌توانید گزینه‌ها را به منظور محدود کردن یا گسترش خروجی `wc` ارسال کنید، همانطور که در جدول 8.3 خلاصه شده است. از بین گزینه‌های جدول 8.3، `-w` و `-c` به عنوان پیش‌فرض استفاده می‌شوند، بنابراین تایپ کردن `wc file.txt` معادل `lwc file.txt` است. صفحه راهنمای برنامه چند گزینه دیگر را شرح می‌دهد، اما احتمالاً از گزینه‌های موجود در جدول 8.3 استفاده خواهد کرد.

Table 8.3 Common wc options

Option (long form)	Option (short form)	Description
--bytes	-C	نمایش تعداد بایت فایل
--chars	-M	نمایش تعداد کاراکترهای فایل
--lines	-L	نمایش تعداد خطوط جدید در فایل
--words	-W	نمایش تعداد کلمات در فایل
--max-linelength	-L	نمایش طول بلندترین خط در فایل

نکته: برخی از فایل‌های متنی از رمزگذاری چند بایتی استفاده می‌کنند، به این معنا که یک کاراکتر ممکن است بیش از یک بایت را اشغال کند. بنابراین، گزینه‌های `C` و `M` ممکن است نتایج یکسانی تولید نکنند، اگرچه اغلب این اتفاق می‌افتد.

به یاد داشته باشید که `wc` بر روی فایل‌های متنی ساده به درستی کار می‌کند، اما ممکن است نتایج نادرست یا حتی بی‌معنی را برای فایل‌های متنی با قالب‌هایی مانند HTML یا فایل‌های پردازنده کلمات تولید کند. بهتر است از یک واژه‌پرداز یا ادیتور تخصصی دیگر برای یافتن تعداد کلمات و آمارهای دیگر برای اینگونه فایل‌ها استفاده کنید.

## Using cut

هنگام استخراج داده، `grep` در استخراج خطوط فایل کامل (رکوردها) مفید است. گاهی اوقات، با این حال، شما فقط بخش‌هایی از یک رکورد فایل را نیاز دارید. دستور `cut` می‌تواند در این مورد کمک کند. این دستور متن را از فیلهای مختلف در یک رکورد فایل استخراج می‌کند. اغلب از آن برای استخراج اطلاعات متغیر از یک فایلی که محتويات آن بسیار الگودار هستند استفاده می‌شود.

جدول 8.4 گزینه‌هایی را نشان می‌دهد که با دستور `cut` احتمالاً بیشتر احتیاج خواهد داشت. برای گزینه‌ها و توضیح کامل‌تر از دستور `cut` به صفحه `man` آن مراجعه کنید.

Table 8.4 Common cut options

Option (long form)	Option (short form)	Description
--characters	-c	انتخاب شدن فقط موقعیت(های) نشانگر مشخص
--delimiter	-d	از حائل میدان مشخص شده به عنوان جداکننده میدان استفاده می کند به جای جداکننده پیش فرض (tab)
--fields	-f	انتخاب میدان های مشخص شده را انجام می دهد.
--onlydelimited	-S	خطوطی که دارای جدا کننده نیستند چاپ نمی شوند (پیش فرض چاپ زمانی است که گزینه -f استفاده می شود).

برای استفاده از `cut`, یک یا چند گزینه را به آن منتقل می کنید که مشخص می کند کدام اطلاعات را می خواهید، و سپس یک یا چند نام فایل. به عنوان مثال، دایرکتوری های خانه کاربران در ششمین فیلد که با دو نقطه در فایل `/etc/passwd` جدا شده و نمایش داده می شود. بنابراین، برای استخراج تنها نام دایرکتوری، این دستور را صادر کنید:

```
$ cut -f 6 -d ":" /etc/passwd
```

می توانید نتایج دستور `cut` را با استفاده از تغییر مسیر خروجی به یک فایل ذخیره کنید. تغییر مسیر خروجی در بخش "Redirecting Input and Output" این فصل توضیح داده شده است.

نکته: وقتی از دستور `cut` استفاده می کنید، اطلاعات استخراج شده روی صفحه نمایش نشان داده می شود. با این حال، توجه داشته باشید که فایل(های) مشخص شده تغییر نمی کند.

## Using sort

هنگام کار با مقدار زیادی داده، توانایی مرتب سازی آنها اغلب مفید است. دستور `sort` دقیقاً همین کار را انجام می دهد. با این حال، شما باید از ویژگی های آن آگاه باشید تا به نتایج دلخواه برسید.

برای یک لیست داده ساده که فقط شامل کلمات است، می توانید از دستور `sort` بدون هیچ گزینه ای برای مرتب سازی آن به ترتیب حروف الفبا استفاده کنید، همان طور که در اینجا نشان داده شده است:

```
$ cat pets.txt
```

```
fish
```

```
cat
```

```
dog
```

```
bird
```

```
$ sort pets.txt
```

```
bird
```

```
cat
```

```
dog
```

```
fish
```

نکته: همانند دستور `cut`, هنگام استفاده از دستور `sort` هیچ تغییری در داده‌های فایل ایجاد نمی‌شود. فقط خروجی مرتب می‌شود.

ممکن است هنگام مرتبسازی داده‌های عددی نیاز به استفاده از گزینه‌ها داشته باشید تا به نتایج دلخواه برسیید. مثال در شکل 8.1 نشان می‌دهد که چه اتفاقی می‌افتد وقتی از دستور `sort` برای مرتبسازی یک لیست داده عددی استفاده می‌شود. اعداد تا زمانی که گزینه `n`-استفاده نشود، به درستی به ترتیب عددی مرتب نمی‌شوند.

The screenshot shows a terminal window titled "christine@Kaylee-FF ~". The window contains the following command-line session:

```
$ cat numbers.txt
7
42
3
1138
$
$ sort numbers.txt
1138
3
42
7
$
$ sort -n numbers.txt
3
7
42
1138
$
```

Figure 8.1 Sorting a numeric data list

شکل 8.1 اهمیت استفاده از گزینه مناسب دستور sort را برای دستیابی به نتایج دلخواه نشان می‌دهد. چند گزینه پرطرفدار دستور sort در جدول 8.5 فهرست شده‌اند.

Table 8.5 Common sort options

Option (long form)	Option (short form)	Description
--dictionaryorder	-d	فقط فضاهای خالی و کاراکترهای الفبایی-عددی را در نظر می‌گیرد؛ کاراکترهای خاص را در نظر نمی‌گیرد.
--ignorecase	-f	حروف بزرگ و کوچک را نادیده می‌گیرد (پیش‌فرض این است که حروف بزرگ و کوچک را در نظر بگیرد و حروف بزرگ را ابتدا قرار دهد).

--numericsort	-n	بر اساس مقدار عددی رشته‌ای مرتب می‌کند.
--output=file	-o	نتایج را به فایل مشخص شده می‌نویسد.
--reverse	-r	مرتب‌سازی را به صورت نزولی انجام می‌دهد (پیش‌فرض این است که به صورت صعودی مرتب کند).

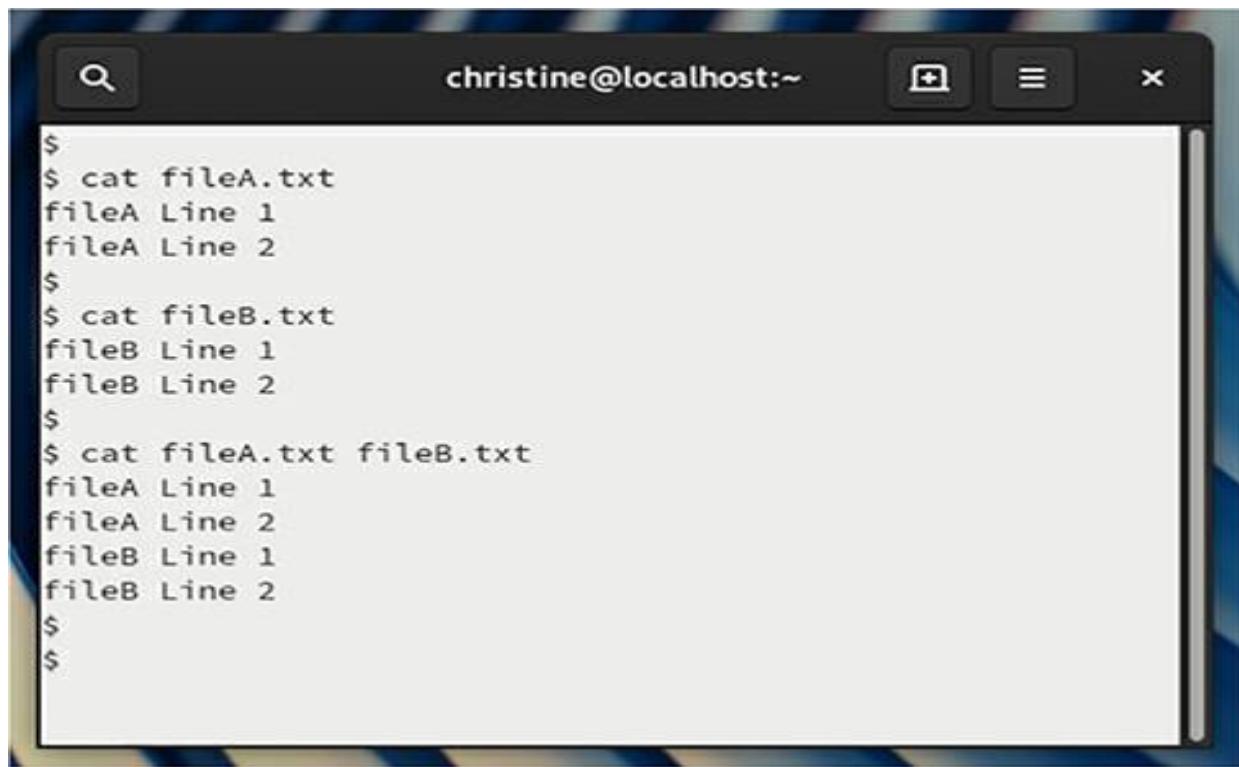
علاوه بر گزینه‌هایی که در جدول 8.5 لیست شده‌اند، بسیاری از گزینه‌های مفید دیگر برای دستور sort وجود دارند. برای کاوش در مورد گزینه‌های دیگر، به صفحات man برای دستور sort مراجعه کنید.

## Using cat

فصل 5، "Getting to Know the Command Line"، دستور cat را معرفی کرد. اگرچه cat معمولاً برای نمایش فایل‌های متنی کوتاه روی صفحه نمایش استفاده می‌شود، اما می‌تواند فایل‌ها را به هم پیوند دهد. مثال در شکل 8.2 نشان می‌دهد که این دو استفاده را در عمل نشان می‌دهد.

در شکل 8.2 توجه کنید که وقتی دستور cat با یک نام فایل به عنوان آرگومان صادر می‌شود، محتوای فایل روی صفحه نمایش نشان داده می‌شود. اما وقتی دو فایل به عنوان آرگومان استفاده می‌شود، محتوای فایل‌ها به یکدیگر پیوسته می‌شود. خود فایل‌ها تغییر نمی‌کنند؛ تنها خروجی به هم پیوند می‌شود. این می‌تواند بسیار مفید باشد. بخش "Redirecting Input and Output" در این فصل جزئیات نحوه حفظ این خروجی برای استفاده در آینده را شرح می‌دهد.

نکته: برای نمایش تنها 10 خط اول یک فایل، به جای دستور cat از دستور head استفاده کنید. برای مشاهده تنها 10 خط آخر یک فایل، دستور tail مفید است. برای پیدا کردن نحوه نمایش تعداد کمتر یا بیشتری از خطوط فایل با این دستورها، صفحات man آنها را بررسی کنید.



```
christine@localhost:~$ cat fileA.txt
fileA Line 1
fileA Line 2
$
$ cat fileB.txt
fileB Line 1
fileB Line 2
$
$ cat fileA.txt fileB.txt
fileA Line 1
fileA Line 2
fileB Line 1
fileB Line 2
$
```

Figure 8.2 Using cat to display and concatenate files

## Redirecting Input and Output

اگر می‌خواهید خروجی یک برنامه را برای مرجع آینده ذخیره کنید، می‌توانید آن را به یک فایل هدایت کنید. همچنین می‌توانید ورودی را به یک برنامه از یک فایل هدایت کنید. گرچه هدایت ورودی ممکن است عجیب به نظر برسد، برخی برنامه‌ها برای فعال کردن پردازش داده‌ها، مانند فایل‌های متنی خام که از طریق یک برنامه به جستجوی الگوها در متن هدایت می‌شوند، به این ویژگی وابسته‌اند. علاوه بر هدایت خروجی به فایل‌ها یا ورودی از فایل‌ها، می‌توانید خروجی یک برنامه را به عنوان ورودی برنامه دیگری ارسال کنید. یک تکنیک مرتبط شامل دستور `xargs` است، که به شما این امکان را می‌دهد که گزینه‌های کامند لاین را از فایل‌ها یا خروجی برنامه‌های دیگر تولید کنید.

## Using Basic Redirection Operators

هدایت از طریق اپراتورهای هدایتی صورت می‌گیرد که رشته‌های کوتاهی هستند که پس از دستور و آرگومان‌های آن ظاهر می‌شوند. جدول 8.6 پرکاربردترین اپراتورهای هدایت را نشان می‌دهد. توجه داشته باشید که خروجی به دو نوع است:

خروجی استاندارد: این برای پیام‌های عادی برنامه است.

خطای استاندارد: این حاوی پیام‌های خطای است.

داشتن دو نوع خروجی این امکان را فراهم می‌کند که آن‌ها را جدا کرد تا پیام‌های خطای برنامه‌هایی را که ممکن است انتظار داشته باشند نوع خاصی از ورودی را از یک برنامه دیگر دریافت کنند، گیج نکنند.

Table 8.6 Common redirection operators

Effect	Redirection operator
یک فایل جدید حاوی خروجی استاندارد ایجاد می‌کند. اگر فایل مشخص شده وجود داشته باشد، بازنویسی می‌شود.	>
خروجی استاندارد را به فایل موجود اضافه می‌کند. اگر فایل مشخص شده وجود نداشته باشد، آن ایجاد می‌شود.	>>
یک فایل جدید حاوی پیام‌های خطای استاندارد) ایجاد می‌کند. اگر فایل مشخص شده وجود داشته باشد، بازنویسی می‌شود.	2>
خطای استاندارد را به فایل موجود اضافه می‌کند. اگر فایل مشخص شده وجود نداشته باشد، آن ایجاد می‌شود.	2>>
یک فایل جدید حاوی هم خروجی استاندارد و هم خطای استاندارد ایجاد می‌کند. اگر فایل مشخص شده وجود داشته باشد، بازنویسی می‌شود.	&>
محتوای فایل مشخص شده را به عنوان ورودی استاندارد برای استفاده ارسال می‌کند.	<
متن را در خطوط بعدی به عنوان ورودی استاندارد قبول می‌کند.	<>
باعث می‌شود فایل مشخص شده برای هر دو ورودی استاندارد و خروجی استاندارد استفاده شود.	>>

به عنوان یک مثال از هدایت خروجی، یک دستور grep را برای جستجوی اطلاعاتی در مورد یک کاربر خاص در تمام فایل‌های پیکربندی در ساختار درختی /etc/ در نظر بگیرید. بدون هدایت، یک دستور اینگونه ممکن است به شکل زیر باشد:

```
# grep -r christine /etc/
[...]
/etc/group:adm:x:4:syslog,christine
/etc/group:cdrom:x:24:christine
/etc/group:sudo:x:27:christine
/etc/group:dip:x:30:christine
/etc/group:plugdev:x:46:christine
[...]
```

چنین خروجی ممکن است بسیار طولانی باشد و ممکن است بعداً می‌خواهید آن را بخوانید. برای انجام این کار، می‌توانید خروجی را به یک فایل هدایت کنید، مانند مثال زیر:

```
# grep -r christine /etc/ > christine-in-etc.txt
```

نکته: هنگام استفاده از اپراتور < مراقب باشید. اگر فایل از قبل وجود داشته باشد، این اپراتور محتوای فعلی فایل را بازنویسی خواهد کرد.

استفاده از اپراتور < خروجی را از دستور grep گرفته و آن را در یک فایل به نام christine-in-etc.txt قرار می‌دهد. اگر می‌خواهید خروجی را ببینید، از دستور less استفاده کنید:

```
# less christine-in-etc.txt
[...]
/etc/group:adm:x:4:syslog,christine
/etc/group:cdrom:x:24:christine
/etc/group:sudo:x:27:christine
/etc/group:dip:x:30:christine
/etc/group:plugdev:x:46:christine
```

در این مثال، نسبت به فقط تایپ کردن دستور /etc/grep -r christine، زیاد بهره‌ای نبردهاید، اما ممکن است در موارد دیگر بهره‌مند شوید. به عنوان مثال، فرض کنید یک دستور چندین پیام خط‌تولید می‌کند، در این صورت می‌توانید خط‌ای استاندارد را به یک فایل هدایت کرده و رشته‌هایی که ممکن است مرتبط باشند را جستجو کنید، حتی در حالی که سعی می‌کنید دستور را اجرا کنید، یا نسخه‌ای از آن را دوباره اجرا کنید.

در این مثال بعدی نشان داده می‌شود که پیام‌های خط و پیام‌های عادی برنامه مجزا هستند. اگر به عنوان یک کاربر عادی دستور /etc/grep -r christine را تایپ کنید (نام کاربری خود را به جای christine جایگزین کنید)،

احتمالاً خروجی‌ای مانند آنچه قبلًا نشان داده شد را خواهید دید، که فایل‌هایی را که نام کاربری شما در آنها ظاهر شده است مشخص می‌کند؛ با این حال، همچنین ممکن است پیام‌های خطا را ببینید، زیرا شما اجازه خواندن برخی از فایل‌ها در /etc را ندارید:

```
grep: /etc/cups/subscriptions.conf.0: Permission denied
```

```
grep: /etc/security/opasswd: Permission denied
```

اطلاعات در مورد فایل‌هایی که نام کاربری christine در آنها ظاهر شده است از طریق خروجی استاندارد نمایش داده می‌شود، اما خطاهای از طریق خطای استاندارد نمایش داده می‌شوند.

نکته: تنها تفاوت بین هدایت خروجی استاندارد با استفاده از اپراتور هدایت < و هدایت خطای استاندارد با استفاده از اپراتور > 2، عدد 2 است. این به این دلیل است که عدد 2 نمایانگر خطای استاندارد در کامند لاین است.

اگر به خطاهای علاقه‌ای ندارید، می‌توانید آنها را به /dev/null هدایت کنید، یک فایل دستگاهی که به عنوان سطل زباله برای داده‌هایی که می‌خواهید دور بربیزید، استفاده می‌شود:

```
$ grep -r christine /etc/ 2> /dev/null
```

به همین ترتیب، اگر خروجی استاندارد را به یک فایل هدایت کنید اما خطای استاندارد را هدایت نکنید، پیام‌های خطا را روی صفحه‌ی نمایش خود مشاهده خواهید کرد. با این حال، فایلی که ایجاد می‌کنید (مانند christine-in-etc.txt از دستور قبلی) شامل پیام‌های خطا نخواهد بود. ممکن است بخواهید همه انواع مختلف هدایت خروجی را با استفاده از دستور grep -r christine /etc/ (با جایگزینی نام کاربری خود به جای christine) را امتحان کنید تا احساسی از نحوه کار آنها بدست آورید.

## Using Pipes

نوع دیگری از هدایت خروجی، کامند لاین لوله یا پایپ‌لاین است. در یک پایپ، خروجی استاندارد از یک برنامه به عنوان ورودی استاندارد به برنامه دوم هدایت می‌شود. شما با استفاده از یک خط عمودی () بین دو دستور یک پایپ ایجاد می‌کنید؛ این کلید معمولاً بالای کلید Enter روی صفحه کلید قرار دارد و با Shift قابل دسترسی است. پایپ‌لاین‌ها زمانی که به روش‌های مختلف استفاده شوند، می‌توانند مفید باشند. برای مثال، ممکن است خروجی طولانی یک برنامه را از طریق دستور less هدایت کنید، که به شما امکان می‌دهد خروجی را به بالا و پایین صفحه‌بندی کنید. در این مثال، دستور cut دایرکتوری‌های خانگی کاربران را از فایل /etc/passwd می‌کشد و سپس آن را به عنوان ورودی به دستور less هدایت می‌کند (که در فصل 5 توضیح داده شده است) برای مشاهده آسان‌تر نتایج:

```
$ cut -f 6 -d ":" /etc/passwd | less
```

نکته: فایل /etc/passwd و حساب‌های کاربری به طور مفصل در فصل 12، "Understanding Basic Security" پوشش داده شده‌اند.

اگلباً از grep در پایپ‌لاین‌ها برای جستجوی کلمات کلیدی در خروجی استفاده می‌شود. در این مثال، دستور cut شیوه‌ای پیش‌فرض کاربران را از فایل /etc/passwd / استخراج می‌کند و سپس آن را به عنوان ورودی به دستور grep هدایت می‌کند تا به دنبال کلمه کلیدی bash بگردد:

```
$ cut -f 7 -d ":" /etc/passwd | grep bash
```

شما محدود به استفاده از یک پایپ در کامند لاین خود نیستید. به عنوان مثال، می‌توانید مثال قبلی را با اضافه کردن یک پایپ دوم مفیدتر کنید. برای تعیین تعداد کاربرانی که در رکورد /etc/passwd خود شل دارند، از یک پایپ دیگر استفاده کنید و دستور wc را به انتهای پایپ‌لاین اضافه کنید:

```
$ cut -f 7 -d ":" /etc/passwd | grep bash | wc -l
```

237

مثال قبلی شامل سه دستور در پایپ‌لاین بود. در این سیستم، 237 حساب کاربری وجود دارند که در رکورد /etc/passwd خود شل دارند.



### Real World Scenario

#### Creating Custom Tools

هنگام استفاده از دستورات فیلتر مانند cut، grep و wc با پایپ‌ها و هدایت‌ها، می‌توانید ابزارهای سفارشی قدرتمندی ایجاد کنید. با این ابزارها، می‌توانید بازبینی‌های امنیتی انجام دهید، گزارش‌های استفاده ایجاد کنید، فایل‌های لاغ مبتنی بر متن را تحلیل کنید و غیره. محدودیت شما فقط دانش کامند لاین و تخیل شما است.

در فصل 11، "Creating Scripts"، خواهید آموخت که چگونه برنامه‌هایی برای خودکارسازی بسیاری از وظایف ضروری و روتین مدیریت سیستم ایجاد کنید. وقتی بفهمید چگونه از این ابزارها در کامند لاین استفاده کنید، می‌توانید بعداً به سرعت آن‌ها را درون اسکریپت‌های شل به کار بگیرید.

## Generating Command Lines

پایپ‌لاین‌ها می‌توانند برای وظایف پیچیده مفید باشند. به عنوان مثال، فرض کنید می‌خواهید هر فایل در یک ساختار دایرکتوری را که نام آن با تیلدا (~) ختم می‌شود حذف کنید. (این نام فایل نشان‌دهنده فایل‌های پشتیبانی است که توسط برخی ویرایشگرهای متن ایجاد می‌شود). با یک ساختار دایرکتوری بزرگ، این وظیفه می‌تواند دلهره‌آور باشد. دستور معمول حذف فایل (rm)، که به تفصیل در فصل 7 توضیح داده شده است) گزینه‌ای برای جستجو و حذف هر فایل در ساختار دایرکتوری که با چنین معیار خاصی مطابقت داشته باشد، ارائه نمی‌دهد. دستور find می‌تواند بخش جستجو را انجام دهد (که قبلاً به طور مفصل توضیح داده شده است)، اما بخش حذف را نه.

راه حل این است که خروجی دستور `find` را برای ایجاد مجموعه‌ای از کامند لاین‌ها با استفاده از `rm` ترکیب کنید. این کار را می‌توان به سه روش اصلی انجام داد:

**xargs**: هدف دستور `xargs` در یک پایپ‌لاین، ساخت یک دستور از ورودی استاندارد آن است. دستور اصلی برای این دستور به صورت زیر است:

```
xargs [OPTIONS] [COMMAND [INITIAL-ARGUMENTS]]
```

دستوری است که می‌خواهید اجرا کنید، و `INITIAL-ARGUMENTS` یک لیست از آرگومان‌هایی هستند که می‌خواهید به دستور منتقل کنید. `OPTIONS`، گزینه‌های `xargs` هستند؛ آنها به دستور منتقل نمی‌شوند. هنگامی که `xargs` را اجرا می‌کنید، برای هر کلمه که به آن در ورودی استاندارد منتقل می‌شود، یک بار دستور اجرا می‌کند، و آن کلمه را به لیست آرگومان‌ها برای دستور اضافه می‌کند. اگر می‌خواهید چندین گزینه را به دستور منتقل کنید، می‌توانید آنها را با قرار دادن آنها درون علامت نقل قول محافظت کنید.

برای مثال، وظیفه‌ی حذف همه آن فایل‌های پشتیبان را، که با کاراکتر `~` مشخص شده‌اند، در نظر بگیرید. شما می‌توانید این کار را انجام دهید با اتصال خروجی دستور `find` به `xargs` که بعداً دستور `rm` را فراخوانی می‌کند:

```
$ find . -name "*~" | xargs rm
```

بخش اول این پایپ‌لاین (`./ -name ~*`) تمام فایل‌های موجود در دایرکتوری فعلی (`./`). یا زیردایرکتوری‌های آن را که با یک تیلدا (`~`) ختم می‌شوند، پیدا می‌کند. سپس این لیست به `xargs` پیشنهاد می‌شود، که هر فایل پیداشده را به دستور `rm` خود اضافه می‌کند. وظیفه‌ی پیچیده پیدا کردن و حذف تمام فایل‌های پشتیبان به این شکل انجام می‌شود.

نقطه بازگشته‌ی یک ابزار مشابه با `xargs` در بسیاری از جنبه‌ها است که یک کاراکتر به چپ از کلید شماره 1 در اکثر کیبوردها است. نقطه بازگشته با کاراکتر تک نقطه (`'`) که بر روی کیبورد در سمت راست و نزدیک به علامت نقطه ویرگول (`;`) قرار دارد، یکسان نیست.

متن قرار داده شده در دو علامت نقطه بازگشته به عنوان یک دستور جداگانه در کامند لاین در نظر گرفته می‌شود و نتایج آن در کامند لاین جایگزین می‌شود. به عنوان مثال، برای حذف آن فایل‌های پشتیبان، دستور زیر را تایپ می‌کنید:

```
$ rm `find . -name "*~" `
```

استفاده از نقطه بازگشته به این صورت، با نام جایگزینی دستور، به دلیل ارائه جایگزین برای یکی از آرگومان‌های دستور، جایگزینی دستور نامیده می‌شود. در مثال قبلی، نتایج دستور `find` جایگزین نام فایل به عنوان آرگومان دستور `rm` گرفته شد.

قالب \$(): به این دلیل که بسیار آسان است که نقطه‌بازگشتنی (\$) را با علامت نقل قول تکی ('') اشتباه گرفت، استفاده از نقطه‌بازگشتنی برای جایگزینی دستور به طور کلی از چشم می‌افتد. اگر این قالب در توزیع شما موجود باشد، بهتر است از یک فرم دیگر از جایگزینی دستور، یعنی قالب \$() استفاده کنید:

```
$ rm $(find . -name "*~")
```

نتایج با استفاده از این قالب جایگزینی دستور مشابه هستند و به طور کلی به دلیل پرانترهای که دستور اول را به اجرا می‌گیرند و نه نقطه‌بازگشتهای کوچک (که به نقل قول شبیه هستند)، خواندن آنها راحت‌تر است.

## Archiving Data

یک ابزار آرشیو فایل یک گروه از فایل‌ها را در یک "پکیج" فایل تجمعی می‌کند که می‌توانید آن را به راحتی در یک سیستم تکی حرکت دهید؛ آن را به یک فلاش درایو USB یا سایر رسانه‌های قابل جابجایی نسخه‌برداری کنید؛ یا آن را از طریق یک شبکه منتقل کنید. لینوکس از چندین دستور آرشیوسازی پشتیبانی می‌کند، معروف‌ترین آنها tar و zip است. علاوه بر درک این دستورات، باید با پیامدهای استفاده از فشرده‌سازی با آنها آشنا باشید.

نکته: یک برنامه آرشیو دیگر به نام cpio در بعضی موارد در لینوکس استفاده می‌شود. این برنامه از نظر اصلی مشابه tar است، اما در جزئیات عملیاتی متفاوت است.

## Using tar

برنامه tar (tape archiver) توسعه یافته برای آرشیو کردن است. بدون توجه به نامش، می‌توانید از tar برای پشتیبان‌گیری (یا همان آرشیو کردن) از داده‌ها به دیسک سخت یا رسانه‌های دیگر استفاده کنید، نه فقط به آنها. برنامه tar یک ابزار محبوب برای آرشیو کردن فایلهای داده‌ای مختلف به یک فایل واحد، که به فایل آرشیو (فایلهای اصلی بر روی دیسک شما باقی می‌مانند) خوانده می‌شود، می‌باشد. به دلیل اینکه فایل آرشیو حاصل ممکن است بسیار بزرگ باشد، اغلب به صورت فشرده از طریق برنامه tar به یک tarball تبدیل می‌شود. در واقع، tarball‌ها اغلب برای انتقال چندین فایل بین کامپیوترها در یک مرحله استفاده می‌شوند، مانند زمان توزیع سورس کد.

نکته: می‌توانید یک فایل آرشیو tar را با ابزار tar ایجاد کنید و سپس آن را بعداً با استفاده از یک ابزار فشرده‌سازی به یک tarball فشرده کنید. ابزارهای فشرده‌سازی بعداً در این فصل مورد بررسی قرار می‌گیرند.

برنامه tar یک بسته پیچیده با بسیاری از گزینه‌های است، اما بیشتر کارهایی که با این ابزار انجام خواهد داد، با چندین گزینه متداول پوشش داده می‌شود. جدول 8.7 گزینه‌های اصلی tar را فهرست می‌کند، و جدول 8.8 تعیین‌کننده‌ها را فهرست می‌کند که عملکرد گزینه‌ها را بیشتر تغییر می‌دهند. هرگاه tar را اجرا کنید، دقیقاً یک گزینه را استفاده می‌کنید، و معمولاً حداقل یک تعیین‌کننده استفاده می‌کنید.

Table 8.7 tar commands

Option (long form)	Option (short form)	Description
--create	-C	آرشیو ایجاد می‌کند
--concatenate	-A	فایل‌های tar را به یک آرشیو اضافه می‌کند
--append	-r	فایل‌های غیر tar را به یک آرشیو اضافه می‌کند
--update	-u	فایل‌هایی را که جدیدتر از فایل‌های موجود در یک آرشیو هستند، اضافه می‌کند
--diff or --compare	-d	آرشیو را با فایل‌های موجود در دیسک مقایسه می‌کند
--list	-t	محتوای یک آرشیو را فهرست می‌کند
--extract or --get	-X	فایل‌ها را از یک آرشیو استخراج می‌کند

نکته: برخلاف اکثر گزینه‌های تک حرفی در لینوکس، می‌توانید از گزینه‌ها و تعیین‌کننده‌های کوتاه tar بدون خط تیره (-) قبل از آنها استفاده کنید. این به سبک قدیمی معروف است. به عنوان مثال، اگر از یک دستور شامل گزینه‌های cvf استفاده کردید، معمولاً می‌توانید آن را به سبک قدیمی cvf تغییر دهید.

Table 8.8 tar qualifiers

Qualifier (long form)	Qualifier (short form)	Description
--directory dir	-C	به دایرکتوری dir قبل از اجرای عملیات‌ها تغییر پیدا می‌کند
--file [host:]file	-f	از فایلی به نام file در کامپیوتری به نام host به عنوان فایل آرشیو استفاده می‌کند.

--listedincremental file	-g	یک بک آپ یا بازیابی افزایشی با استفاده از file بعنوان لیست فایل های آرشیو شده انجام میدهد
--one-filesystem	(none)	از تنها یک فایل سیستم (پارتیشن) بک آپ یا بازیابی انجام میدهد
--multivolume	-M	چندین آرشیو استخراج یا ایجاد میکند
--tapedlength N	-L	Tape ها را به N کیلوبایت تغییر میدهد
--samepermissions	-p	تمام اطلاعات حفاظتی را حفظ میکند
--absolutenames	-P	/ جلوی نام فایل ها را نگه میدارد
--verbose	-v	تمام فایل های خوانده شده یا استخراج شده را لیست میکند؛ زمانی که با --list استفاده میشود، سایز فایلها، مالکیت، و تاریخ را نشان میدهد
--verify	-W	آرشیو را بعد نوشتن آن تایید میکند
--exclude file	(none)	فایل file را از آرشیو مستثنی میکند
--excludefrom file	-X	فایل های لیست شده در file را از آرشیو مستثنی میکند
--gzip or --ungzip	-Z	یک آرشیو را با gzip پردازش میکند
--bzip2	-j	یک آرشیو را با bzip2 پردازش میکند
--xz	-J	یک آرشیو را با xz پردازش میکند

مثالی از آرشیو و فشردهسازی فایل‌ها برای ایجاد یک tarball در شکل 8.3 نشان داده شده است. فایل‌ها که در دایرکتوری /42home/christine/Project قرار دارند، به یک فلش درایو USB آرشیو می‌شوند. فلش درایو در media/christine/USB20FD/ مونت شده است، همانطور که در شکل 8.3 نشان داده شده است. گزینه‌های -czvf با دستور برای ایجاد (c)، فشردهسازی با استفاده از gzip یا (z)، نمایش فایل‌های در حال آرشیو شدن (v)، و ایجاد یک فایل آرشیو (f) استفاده می‌شوند.

```

christine@Kaylee-FF ~
File Edit View Search Terminal Help
$ ls -l /home/christine/Project42/
total 24
-rw-rw-r-- 1 christine christine 12 Jul 23 18:43 DevOps-WeeklyMeeting.odt
-rw-rw-r-- 1 christine christine 9948 Jul 23 18:42 P42-Data.txt
-rw-rw-r-- 1 christine christine 2487 Jul 23 18:44 TeamMemberContactInfo.odt
-rw-rw-r-- 1 christine christine 2487 Jul 23 18:42 timeline.odt
$ tar -czvf /media/christine/USB20FD/P42.tgz /home/christine/Project42/
tar: Removing leading '/' from member names
/home/christine/Project42/
/home/christine/Project42/timeline.odt
/home/christine/Project42/DevOps-WeeklyMeeting.odt
/home/christine/Project42/TeamMemberContactInfo.odt
/home/christine/Project42/P42-Data.txt
$ ls -l /media/christine/USB20FD/P42.tgz
-rwxrwxrwx 1 christine christine 1345 Jul 23 18:52 /media/christine/USB20FD/P42.tgz
$ 
$ 

```

Figure 8.3 Creating an archive tarball

توجه داشته باشید که در شکل 8.3، کاراکتر / اولیه به طور خودکار از نام فایل‌های آرشیو شده (که در شکل به عنوان نام اعضا خوانده می‌شود) حذف می‌شود. این امر به شما امکان می‌دهد که فایل‌ها را به راحتی به مکانی جدید استخراج کنید. برای مثال، اگر فلاش درایو از شکل 8.3 را به سیستم دیگری منتقل کنید، آن را مونت کنید، فایل tarball با نام P42.tgz را به یک دایرکتوری کپی کنید و بخواهید آرشیو را استخراج کنید، می‌توانید با دستور دیگری این کار را انجام دهید:

```
$ tar -xzvf P42.tgz
```

این دستور یک زیردایرکتوری به نام 42home/christine/Project در دایرکتوری کاری فعلی ایجاد کرده و آن را با فایل‌های tarball پر می‌کند. توجه کنید که تنها یک تغییر در گزینه‌های دستور tar نسبت به گزینه‌های استفاده شده در شکل 8.3 لازم است: گزینه c به x برای استخراج فایل‌ها تغییر کرده است.

**نکته:** برنامه tar اطلاعات مالکیت و مجوزهای لینوکس را حتی زمانی که آرشیو بر روی یک فایل سیستمی ذخیره می‌شود که اینگونه فایل‌های پیوسته را پشتیبانی نمی‌کند، حفظ می‌کند.

اگر نمی‌دانید چه چیزی در یک آرشیو است، معمولاً بهتر است قبل از استخراج محتوا با دستور `-list` آن را بررسی کنید. اگرچه عمل متداول ایجاد tarball است که فایل‌ها را در یک زیردایرکتوری تکی ذخیره می‌کند،

گاهی اوقات tarball‌ها بسیاری از فایل‌ها را در دایرکتوری کاری فعلی رها می‌کنند که این می‌تواند آنها را سخت برای پیدا کردن کند اگر دستور را در یک دایرکتوری اجرا کنید که از پیش بسیاری از فایل‌ها دارد.

## Using Compression

در لینوکس، برنامه‌های 2gzip، bzip و xz همه فایل‌ها را فشرده می‌کنند. به عنوان مثال، ممکن است یک فایل گرافیکی بزرگ را به این صورت فشرده کنید:

```
$ xz biggraphics.tiff
```

نتیجه یک فایل با نام مشابه به فایل اصلی اما با یک پسوند نام جدید برای شناسایی آن به عنوان یک فرمت فشرده است. در این حالت خاص، نتیجه به صورت biggraphics.tiff.xz خواهد بود.

بیشتر برنامه‌های گرافیکی فایل‌هایی که به این شکل فشرده شده‌اند را نمی‌خوانند. بنابراین، برای استفاده از یک فایل که فشرده شده است، باید آن را با یک برنامه Matching فشرده کنید. برای فشرده سازی فایل که در مثال قبلی فشرده شده است، از این دستور استفاده کنید:

```
$ unxz biggraphics.tiff.xz
```

جدول 8.9 خلاصه‌ای از برنامه‌های فشرده‌سازی، برنامه‌های متناظر آنها برای بازکردن فشرده‌سازی، و پسوندهای نام فایلی است که آنها ایجاد می‌کنند را مشخص می‌کند.

Table 8.9 Compression and uncompression programs and filename extensions

Compression program	Uncompression program	Filename extension
gzip	gunzip	.gz
bzip2	bunzip2	.bz2
xz	unxz	.XZ

نکته: به طور کلی، gzip کمترین فشرده‌سازی را ارائه می‌دهد و xz بیشترین فشرده‌سازی را. به همین دلیل است که هسته لینوکس در حال حاضر با xz فشرده شده است.

برنامه tar پشتیبانی صریح از همه سه استاندارد فشرده‌سازی فوق را فراهم می‌کند، و tarballs اغلب دارای پسوندهای نام فایل منحصر به فرد خود هستند که نشان دهنده فشرده‌سازی استفاده شده است:

نکته: گاهی اوقات دو پسوند بر روی یک فایل tarball استفاده می‌شود، که به یک نام فایل مانند my-work.tar.b2 منجر می‌شود.

- .tgz برای فشرده شده tarballs
- .tbz یا 2tbz برای فشرده شده tarballs
- .txz برای فشرده شده tarballs

زمانی که از فشرده سازی برای ایجاد یک tarball استفاده می کنید (با استفاده از گزینه های z، Z یا L برای tar)، برنامه فشرده سازی بر روی کل tarball با تمامی فایل هایش عمل می کند به جای فشرده سازی هر فایل به صورت جداگانه درون tarball. این می تواند نسبت به فشرده سازی هر فایل به صورت جداگانه و سپس گروه بندی آنها با هم، نسبت فشرده سازی را بهبود بخشد. با این حال، این کار را اگر فایل آسیب ببیند، استخراج داده ها از یک فایل سخت تر می شود.

نکته: به طور معمول، فایل های متن ساده به خوبی فشرده می شوند، فایل های با این روش به خوبی متوسط فشرده می شوند، و داده های پیش فشرده (مانند اکثر فرمت های فایل ویدیو) به طور ضعیف فشرده می شوند یا حتی ممکن است در اندازه زمانی که دوباره فشرده می شوند افزایش یابند.

برنامه های فشرده سازی 2gzip، bzip و xz همگی فشرده سازی بدون اتفاف را اعمال می کنند، به این معنی که داده ای که از طریق باز کردن فشرده سازی فایل بازیابی می شود، دقیقاً همان چیزی است که وارد آن شده بود. برخی از فرمت های فایل گرافیکی، صوتی و تصویری فشرده سازی با اتفاف را اعمال می کنند که در آن برخی داده ها حذف می شوند. ابزارهای فشرده سازی با اتفاف هرگز نباید بر روی فایل های برنامه، فایل های پیکربندی سیستم یا بیشتر فایل های داده کاربر استفاده شوند؛ هر گونه اتفاف در چنین فایل هایی می تواند فاجعه بار باشد. به همین دلیل است که tar فقط از ابزارهای فشرده سازی بدون اتفاف پشتیبانی می کند.

## Using zip

در خارج از دنیای یونیکس و لینوکس، فرمت فایل zip یک فرمت رایج است که نقشی مشابه یک tarball فشرده را ایفا می کند. لینوکس فرمان zip را برای ایجاد فایل های zip و ابزار unzip را برای استخراج فایل ها از یک آرشیو zip فراهم می کند. فایل های zip معمولاً پسوند نام فایل.zip دارند.

در بیشتر موارد، می توانید با انتقال نام یک فایل هدف به همراه لیست فایل ها به ابزار، یک آرشیو zip ایجاد کنید:

```
$ zip newzip.zip afile.txt figure.tif
```

این فرمان فایل newzip.zip را ایجاد می کند که شامل فایل های figure.tif و afile.txt است. (فایل های اصلی روی دیسک شما باقی می مانند). در برخی موارد، نیاز خواهد داشت تا از گزینه های zip برای دستیابی به نتایج دلخواه استفاده کنید. جدول 8.10 مهم ترین گزینه های zip را خلاصه می کند؛ اما برنامه از گزینه های بسیاری دیگری نیز پشتیبانی می کند. برای جزئیات بیشتر به صفحه man آن مراجعه کنید.

Table 8.10 Common zip options

Option (long)	Optio	Description
---------------	-------	-------------

form)	n (short form)	
N/A	-0 -9	مقدار فشرده‌سازی را تنظیم می‌کند؛ 0- هیچ فشرده‌سازی اعمال نمی‌کند، 1- فشرده‌سازی کم (اما سریع) اعمال می‌کند، و به همین ترتیب تا 9- که فشرده‌سازی حداکثر (اما کند) را اعمال می‌کند.
--delete	-d	فایل‌های مشخص شده را از فایل آرشیو حذف می‌کند.
--encrypt	-e	آرشیو را با یک رمز عبور رمزگذاری می‌کند (زیپ از شما این رمز عبور را می‌خواهد).
--freshen	-f	فایل‌ها را در یک آرشیو بهروزرسانی می‌کند اگر از زمان ایجاد آرشیو اصلی تغییر کرده باشند.
--fix or --fixfix	-F or -FF	تعمیرات را بر روی فایل آرشیو آسیب‌دیده انجام می‌دهد. گزینه -fix/-F تعمیرات حداقلی انجام می‌دهد، در حالی که fixfix/-FF به صورت کامل‌تر تعمیر می‌کند.
--filesync	-FS	فایل‌ها را در یک آرشیو بهروزرسانی می‌کند اگر از زمان ایجاد آرشیو اصلی تغییر کرده باشند، و فایل‌ها را از آرشیو حذف می‌کند اگر در سیستم فایل حذف شده باشند.
--grow	-g	فایل‌ها را به یک فایل آرشیو موجود اضافه می‌کند.
--help	-h or -?	اطلاعات کمک اولیه را نمایش می‌دهد.
--move	-m	فایل‌ها را به آرشیو زیپ منتقل می‌کند؛ به عبارت دیگر، فایل‌های اصلی حذف می‌شوند.
--recursepaths	-r	فایل‌ها و زیرشاخه‌ها داخل دایرکتوری‌های مشخص شده را در بر می‌گیرد.
--splitsize size	-s size	یک آرشیو پتانسیلی چند فایلی ایجاد می‌کند، به طوری که هر فایل حداکثر اندازه size بایت باشد. (می‌توانید g، m، k، یا t را به size اضافه کنید تا واحدهای بزرگتر را مشخص کنید.)
--exclude files	-x files	فایل‌های مشخص شده را از آرشیو حذف می‌کند.
--symlinks	-y	نمادگذاری‌های نمادین (به طور معمول، zip فایل‌های متصل شده را شامل می‌شود) را دربرمی‌گیرد.

از بین گزینه‌های جدول 8.10، گزینه ۲- احتمالاً مهمترین است، حداقل اگر می‌خواهید یک درخت کامل از دایرکتوری را فشرده‌سازی کنید. اگر از این گزینه استفاده نکنید، آرشیو شما هیچ زیردایرکتوری‌ای را شامل نخواهد شد. با توجه به سرعت پردازندگان مدرن، استفاده مداوم از ۹- همچنین معقول به نظر می‌رسد تا فشرده‌سازی حداکثری را بدست آورید.

برای برداشتن و استخراج فایل‌ها در یک فایل آرشیو zip، می‌توانید از برنامه unzip استفاده کنید:

```
$ unzip anarchive.zip
```

این مثال فایل‌ها را در فایل anarchive.zip در دایرکتوری فعلی فشرده‌سازی می‌کند. unzip مانند zip از تعداد زیادی گزینه پشتیبانی می‌کند، مهمترین آنها در جدول 8.11 ظاهر شده است.

Table 8.11 Common unzip options

Option	Description
-f	فایل‌ها را از آرشیو به روز می‌کند؛ به عبارت دیگر، فقط فایل‌هایی را که در سیستم فایل اصلی وجود دارند و در آرشیو نسخه‌های جدیدتری از آنها وجود دارد، استخراج می‌کند.
-l	فایل‌ها را در آرشیو لیست می‌کند اما آنها را استخراج نمی‌کند.
-p	فایل‌ها را به یک پایپ لاین استخراج می‌کند.
-t	صحت فایل‌ها را در آرشیو بررسی می‌کند.
-u	فایل‌ها را به روزرسانی می‌کند؛ مشابه -f اما همچنین فایل‌ها را که در سیستم فایل وجود ندارند هم استخراج می‌کند.
-v	فایل‌ها را در آرشیو به یک فرمت بازنویسی بیشتر از آنچه - انجام می‌دهد لیست می‌کند.
-L	نام فایل‌ها را به حروف کوچک تبدیل می‌کند اگر از یک سیستم عامل با حروف بزرگ (مانند DOS) مشتق شده باشند.
-n	فایل‌های موجود را بازنویسی نمی‌کند.
-o	بدون سوال بازنویسی می‌کند.

نکته: فایل‌های زیپ معمولاً شامل فایل‌های "loose" در دایرکتوری اصلی هستند، بنابراین به طور کلی پیشنهاد می‌شود که آرشیوهای zip را در یک زیردایرکتوری خالی که برای این منظور ایجاد می‌کنید، استخراج کنید.

به طور کلی، استفاده از دستور `unzip` بدون هیچ گزینه‌ای به جز نام فایل ورودی به خوبی کار می‌کند. با این حال، ممکن است در برخی مواقع، از یک یا چند گزینه از آن استفاده کنید. گزینه `-l` به ویژه برای بررسی محتوای آرشیو بدون استخراج آن بسیار مفید است.

# CHAPTER 9

## Exploring Processes and Process Data

کامپیوترها دستگاه‌های پویا و چندمنظوره هستند؛ آنها با استفاده از ابزارهای متنوعی از کارها را انجام می‌دهند. این فصل به شرح روش‌های مدیریت این ابزارها می‌پردازد. یک جنبه از مدیریت نرم‌افزار نصب، حذف و ارتقاء بسته‌های نرم‌افزاری است. جنبه دیگر این وظیفه مدیریت برنامه‌ها پس از اجرای آنهاست. در نهایت، این فصل به فایل‌های log می‌پردازد که جزئیات انجام عملکرد برنامه‌های در حال اجرا را ثبت می‌کنند - به ویژه برنامه‌هایی که به طور خودکار و در پس‌زمینه اجرا می‌شوند.

### Understanding Package Management

مدیریت بسته‌ها یک حوزه از لینوکس است که در هر توزیع متفاوت است. با این حال، برخی اصول در بیشتر توزیع‌های لینوکس مشترک هستند. این بخش این اصول را شرح می‌دهد، پس از آن به برخی از مبانی دو سیستم اصلی مدیریت بسته لینوکس می‌پردازد. سپس نحوه مدیریت بسته‌ها با استفاده از هر دو سیستم مدیریت بسته اصلی، مانند RPM Package Manager و سیستم بسته Debian را توضیح می‌دهد.

### Linux Package Management Principles

اگر نرم‌افزاری را در ویندوز نصب کرده باشید، احتمالاً با روش دوبار کلیک بر روی برنامه نصب‌کننده آشنا هستید، که تمام فایل‌های مرتبط با یک برنامه را در مکان مناسب قرار می‌دهد. نصب‌کننده نرم‌افزار ویندوز مشابه یک فایل بسته لینوکس است، اما تعدادی تفاوت وجود دارد. بسته‌های لینوکس دارای ویژگی‌های زیر هستند:

- هر بسته یک فایل تکی است که می‌تواند در یک دیسک ذخیره شود یا از طریق اینترنت منتقل شود.
- فایل‌های بسته لینوکس، بر خلاف نصب‌کننده‌های ویندوز، برنامه نیستند؛ بسته‌ها به برنامه‌های دیگر وابسته‌اند تا کار نصب نرم‌افزار را انجام دهند.
- بسته‌ها شامل اطلاعات وابستگی هستند؛ به عبارت دیگر، می‌توانند به نرم‌افزار بسته‌بندی اطلاع دهنده که برای کار صحیح بسته، چه بسته‌های دیگر یا فایل‌های فردی باید نصب شوند.

نکته: بسیاری از بسته‌های برنامه به بسته‌های کتابخانه وابسته هستند؛ کتابخانه‌ها کدی را فراهم می‌کنند که می‌تواند توسط بسیاری از برنامه‌ها استفاده شود.

- بسته‌ها شامل اطلاعات نسخه هستند تا نرم‌افزار بسته‌بندی بتواند بین دو بسته تشخیص دهد که کدامیک جدیدتر است.
- بسته‌ها شامل اطلاعات معماری هستند تا نوع پردازنده (مانند ARM، x86-64، x86 و غیره) که برای آنها طراحی شده‌اند را تشخیص دهند. یک کد جدأگانه بسته‌هایی را که مستقل از معماری هستند، مانند فونت‌ها و تم‌های دسکتاپ، شناسایی می‌کند.

- بسته‌های باینری (یعنی، آن‌هایی که شامل برنامه‌های اجرایی هستند که برای یک CPU خاص طراحی شده‌اند) معمولاً از بسته‌های سورس (که حاوی سورس کد است که یک برنامه‌نویس می‌تواند درک کند) ساخته می‌شوند. امکان ساخت یک بسته باینری جدید از بسته سورس وجود دارد که در برخی از شرایط غیرعادی مفید است.

نکته: می‌توانید بدون استفاده از ابزار بسته‌بندی، نرم‌افزار را به صورت دستی از سورس کد کامپایل و نصب کنید. این موضوع پیشرفت‌هه، خارج از دامنه این کتاب است.

نرم‌افزار بسته‌بندی، یک پایگاه داده از اطلاعات بسته‌های نصب شده (the package database) را نگهداری می‌کند. این اطلاعات شامل نام‌ها و شماره‌های نسخه تمامی بسته‌های نصب شده، و همچنین مکان‌های تمامی فایل‌های نصب شده از هر بسته می‌باشد. این اطلاعات امکان حذف سریع نرم‌افزار را فراهم می‌کند، تشخیص می‌دهد که آیا وابستگی‌های یک بسته جدید برآورده شده‌اند یا نه، و مشخص می‌کند که آیا یک بسته که در حال تلاش برای نصب آن هستید، قبلًا نصب شده است و اگر بله، آیا نسخه نصب شده قدیمی‌تر از نسخه‌ای که در حال تلاش برای نصب آن هستید است یا خیر.

نکته: بسته‌ها اغلب می‌توانند فایل‌هایی را شامل شوند که به بسیاری از دایرکتوری‌ها در کامپیوتر نصب می‌شوند. این واقعیت باعث می‌شود که پیگیری محتوای بسته بسیار حیاتی باشد.

## Understanding Package Systems

همان‌طور که قبلًا ذکر شد، دو سیستم بسته Debian و RPM، رایج هستند، اگرچه سایر نسخه‌ها نیز وجود دارند. این سیستم‌ها در جزئیات فنی مختلف، همچنین در دستورات استفاده شده برای مدیریت بسته‌ها و در فرمت فایل‌های بسته که استفاده می‌کنند، متفاوت هستند. می‌توانید یک بسته Debian را در یک سیستم مبتنی بر RPM نصب کنید و یا بالعکس. در واقع، حتی نصب یک بسته برای یک توزیع بر روی یک توزیع دیگر هم خطرناک است، حتی زمانی که از همان نوع بسته استفاده می‌شود. این به دلیل این است که یک بسته غیر متعلق به توزیع ممکن است وابستگی‌هایی داشته باشد که با نیازهای بسته‌های لوکال تداخل داشته باشد.

نکته: جدول 1.1 در فصل 1، "Selecting an Operating System"، خلاصه‌ای از برخی ویژگی‌های چندین توزیع محبوب لینوکس را، از جمله سیستم بسته‌بندی هر یک، خلاصه می‌کند.

اصولاً، سیستم‌های بسته ابتدا به صورت محلی کار می‌کردند؛ به این معنا که برای نصب یک بسته در کامپیوترتان، باید ابتدا یک فایل بسته را از اینترنت یا از طریق روش‌های دیگری دانلود می‌کردید. تنها پس از آن می‌توانستید از یک دستور محلی برای نصب بسته استفاده کنید. این روش، هنگامی که یک بسته به تعداد زیادی وابستگی داشته باشد، ممکن است خسته‌کننده باشد؛ ممکن است سعی کنید برای نصب آن، وابستگی‌هایی را بیابید که برآورده نشده‌اند، چندین بسته دیگر را دانلود کنید، بعد متوجه شوید که یک یا چند بسته دیگر نیازمند وابستگی‌هایی هستند که برآورده نشده‌اند، و الی آخر. در زمانی که همه این بسته‌های وابسته را پیدا کرده‌اید، ممکن است نیاز داشته باشید که دوازده یا بیشتر بسته را نصب کنید. بنابراین، توزیع‌های مدرن ابزارهای قابل دسترس شبکه‌ای را فراهم می‌کنند تا به فرایند نصب بسته‌ها کمک کنند. این

ابزارها بر اساس مخازن نرم‌افزاری شبکه وابسته هستند، که از آنجا که ابزارها می‌توانند بسته‌ها را به صورت خودکار دانلود کنند. ابزارهای قابل دسترس شبکه‌ای از یک توزیع به توزیع دیگر متفاوت است، به ویژه در میان توزیع‌های مبتنی بر RPM.

در عمل، فرآیند مدیریت نرم‌افزار در لینوکس شامل استفاده از ابزارهای تکست مود یا اینترفیس گرافیکی برای ارتباط با ریپازیتوری نرم‌افزار است. یک عمل نصب نرم‌افزار معمولی به این شکل است:

1. شما یک دستور برای نصب یک برنامه صادر می‌کنید.

2. نرم‌افزار وابستگی‌های برنامه مشخص شده را پیدا کرده و شما را از هر نرم‌افزار اضافی که باید نصب شود مطلع می‌کند.

3. شما یک تأیید نهایی برای نصب نرم‌افزار را صادر می‌کنید (یا تصمیم می‌گیرید که این کار انجام نشود، در این صورت فرآیند متوقف می‌شود).

4. نرم‌افزار تمامی بسته‌های لازم را دانلود می‌کند.

5. نرم‌افزار تمامی بسته‌ها را نصب می‌کند.

نکته: شما می‌توانید اکثر توزیع‌ها را تنظیم کنید تا به جای یا همراه ریپازیتوری اینترنتی، از رسانه‌های لوکال استفاده کنند.

به روزرسانی نرم‌افزار به یک شیوه مشابه کار می‌کند، با این تفاوت که در اغلب به روزرسانی‌ها نیازی به دانلود بسته‌های وابسته نیست. حذف نرم‌افزار می‌تواند به طور کامل به صورت لوکال انجام شود. بسیاری از توزیع‌ها به طور خودکار در طی زمان با ریپازیتوری خود بررسی می‌کنند و شما را هنگامی که به روزرسانی‌ها در دسترس هستند، آگاه می‌سازند. بنابراین، هنگامی که از شما درخواست می‌شود می‌توانید با کلیک کردن چند دکمه، سیستم خود را به روز نگه دارید. به عنوان یک مثال، شکل 9.1 نشان‌دهنده ابزار به روزرسانی نرم‌افزار در لینوکس مینت 18.3 است که یک فهرست از به روزرسانی‌های موجود را نشان می‌دهد.

نکته: بلافضله پس از نصب یک توزیع، ممکن است متوجه شوید که تعداد زیادی از به روزرسانی‌ها در دسترس است.

مدیریت بسته لزوماً دسترسی روت را در بر می‌گیرد که در فصل 13، "Creating Users and Groups!"، به تفصیل توضیح داده شده است. اگر شما دستورات خودکار را برای به روزرسانی نرم‌افزار دنبال کنید، می‌توانید با وارد کردن رمز عبور روت، یا در برخی از توزیع‌ها رمز عبور عادی‌تان، زمانی که نرم‌افزار به روزرسانی برای آن درخواست می‌کند، سیستم را به روز نگه دارید.

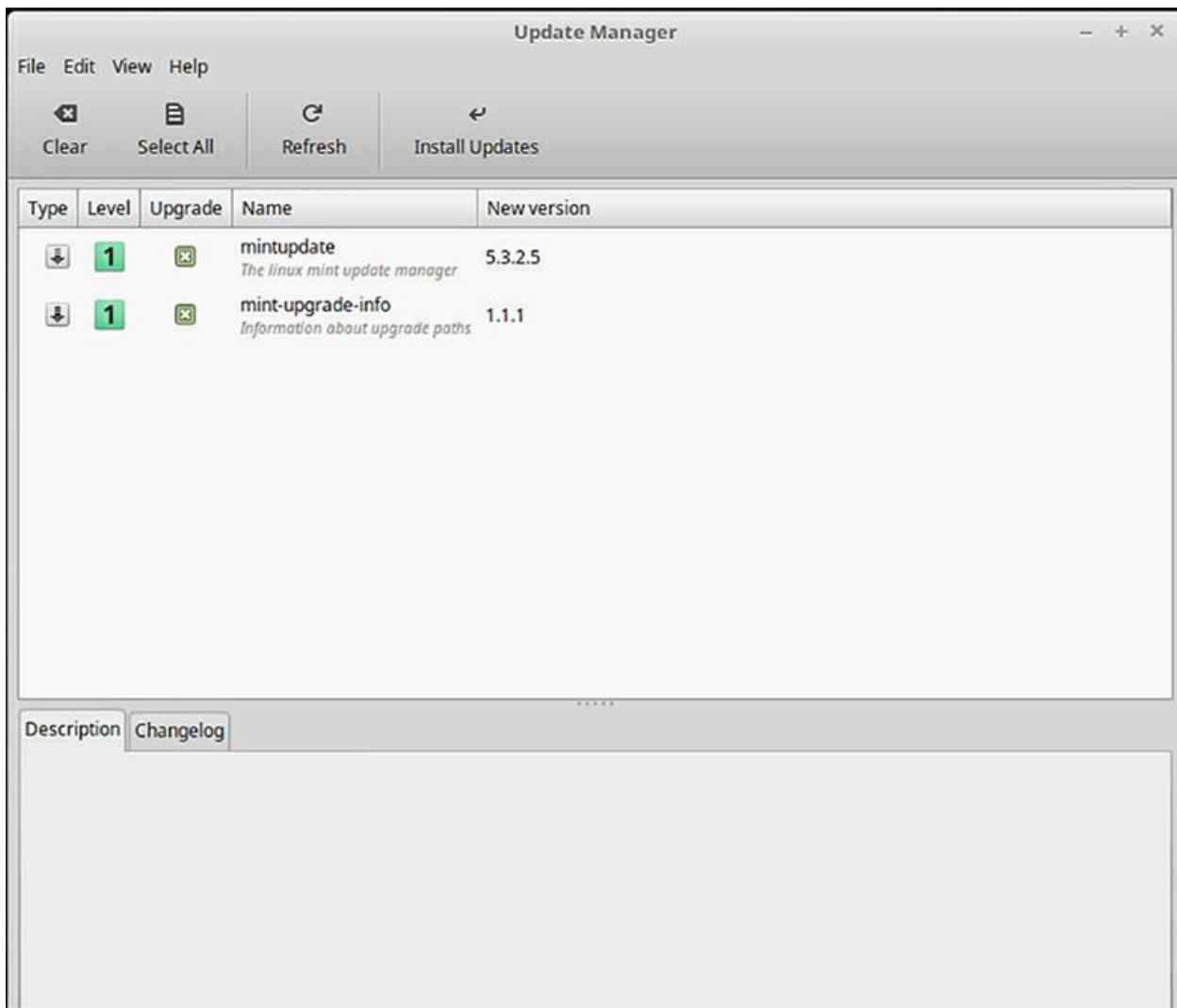


Figure 9.1 Most Linux distributions tell you when updates are available for your software.

## Managing Red Hat Systems

توزیع‌های مبتنی بر RPM شامل رده‌های Red Hat، Fedora، CentOS، SUSE Enterprise، مندربیو و openSUSE هستند. ابزار اصلی برای نصب نرم‌افزار در این توزیع‌ها دستور متنی rpm است. این برنامه بر روی فایل‌های محلی کار می‌کند؛ اما برای استفاده از ریپازیتوری شبکه، باید از یک ابزار دیگر استفاده کنید که بسته به توزیع متفاوت است:

- تکست مود، یا yum یا dnf برای Red Hat، Fedora و CentOS
- آن‌ها استفاده می‌کنند، مانند PackageKit و Yumex
- یا اینترفیس‌های گرافیکی مختلطی برای openSUSE و SUSE Enterprise 2 YaST استفاده می‌کنند.

به دلیل تفاوت‌های بین این توزیع‌ها، به ویژه برای بهروزرسانی‌های قابل دسترسی از شبکه، ارائه توصیف کاملی از تمامی این ابزارها در اینجا غیرممکن است. خوشبختانه، ابزارهای گرافیکی آسان و قابل دسترس هستند. حتی ابزارهای تکست مود نیز به نسبت ساده هستند، اگرچه شما ممکن است نیاز داشته باشید که به صفحات آن‌ها مراجعه کنید تا جزئیات را بیاموزید. به طور معمول، آن‌ها از ساب کامندهای منطقی استفاده می‌کنند، مانند `install` برای نصب یک پسته، مانند:

```
# dnf install yumex
```

ممکن است از این دستور برای نصب ابزار گرافیکی Yumex یا Red Hat، Fedora یا CentOS استفاده کنید. به طور مشابه، می‌توانید یک پسته خاص را با استفاده از ساب کامند حذف (remove) حذف کنید یا تمامی پسته‌های یک کامپیوتر را با استفاده از زیردستور بهروزرسانی (upgrade) بهروزرسانی کنید:

```
# dnf remove zsh
```

```
# dnf upgrade
```

نکته: اگر می‌خواهید هم نرم‌افزار را بهروزرسانی کنید و هم پسته‌ها را حذف کنید، به طور کلی بهتر است ابتدا نرم‌افزار را حذف کنید. این می‌تواند برخی از دانلودها را از بین ببرد و زمان بهروزرسانی را کاهش دهد.

این مثال پسته `zsh` را حذف می‌کند، سپس پسته‌های باقی‌مانده را در سیستم بهروزرسانی می‌کند. هر دو دستور ممکن است تعدادی خط خروجی تولید کنند و ممکن است از شما درخواست شود که عملیات آن‌ها را تأیید کنید. برای کسب اطلاعات بیشتر درباره این ابزار، به صفحه `man yum` (یا هر نرم‌افزار مدیریت پسته دیگری که توزیع شما استفاده می‌کند) مراجعه کنید.

اگر نیاز دارید مستقیماً با فایل‌های پسته RPM برخورد کنید، باید به این نکته توجه کنید که آن‌ها دارای پسوند `.rpm` هستند. این فایل‌ها معمولاً شامل کدهای نوع معماری (مانند `i386` یا `x86_64`) و اغلب کدهای توزیعی که برای آن‌ها طراحی شده‌اند (مانند `30fc` برای Fedora 30) است. به عنوان مثال، `samba-4.10.2-0.fc30.x86_64.rpm` یک فایل پسته برای پسته `samba`، نسخه 4.10.2، نسخه 0، برای Fedora 30، بر روی پلتفرم `x86_64` است. برای نصب آن با استفاده از دستور `rpm`، شما باید عبارت زیر را تایپ کنید:

```
# rpm -Uvh samba-4.10.2-0.fc30.x86_64.rpm
```

این کار نسبت به استفاده از دستور `yum`، به کار نسبتاً بیشتری نیاز دارد زیرا شما باید نام فایل نصب را به طور کامل بدانید.

## Managing Debian Systems

توزیع Debian GNU/Linux سیستم بسته‌های خود را ایجاد کرد و توزیع‌های مبتنی بر Debian مانند اوبونتو و مینت از همان سیستم استفاده می‌کنند. در بالای سیستم بسته اصلی Debian، ابزار پیشرفته بسته (APT) وجود دارد که دسترسی به ریپازیتوری‌های شبکه می‌دهد.

نکته: پیاده‌سازی‌های شخص ثالث APT برای بسیاری از توزیع‌های مبتنی بر RPM نیز وجود دارند. برای جزئیات بیشتر به سایت [apt4rpm.sourceforge.net](http://apt4rpm.sourceforge.net) مراجعه کنید. حداقل یک توزیع مبتنی بر RPM، یعنی PCLinuxOS، به طور بومی از APT استفاده می‌کند.

دستور `dpkg` پایین‌ترین سطح رابط به سیستم بسته‌های Debian است؛ این دستور تقریباً معادل ابزار `rpm` در سیستم‌های مبتنی بر RPM است. همانند ابزار `rpm`، برای استفاده از دستور `dpkg` باید نام دقیق بسته‌ای را که می‌خواهید نصب کنید، بدانید:

```
# dpkg -i samba_4.9.5+dfsg-5+deb10u1_amd64.deb
```

چندین ابزار رابطه‌ای متنی و گرافیکی را در بالای `dpkg` فراهم می‌کنند که مهم‌ترین آن‌ها ابزار متنی `apt-get` یا ابزار جدیدتر `apt` و رابط گرافیکی `Synaptic` هستند. همان‌طور که از نام آن‌ها پیداست، `apt-get` و `Synaptic` دسترسی به ریپازیتوری‌های شبکه از طریق APT را فراهم می‌کنند. شکل 9.2 استفاده از `Synaptic` را نشان می‌دهد.

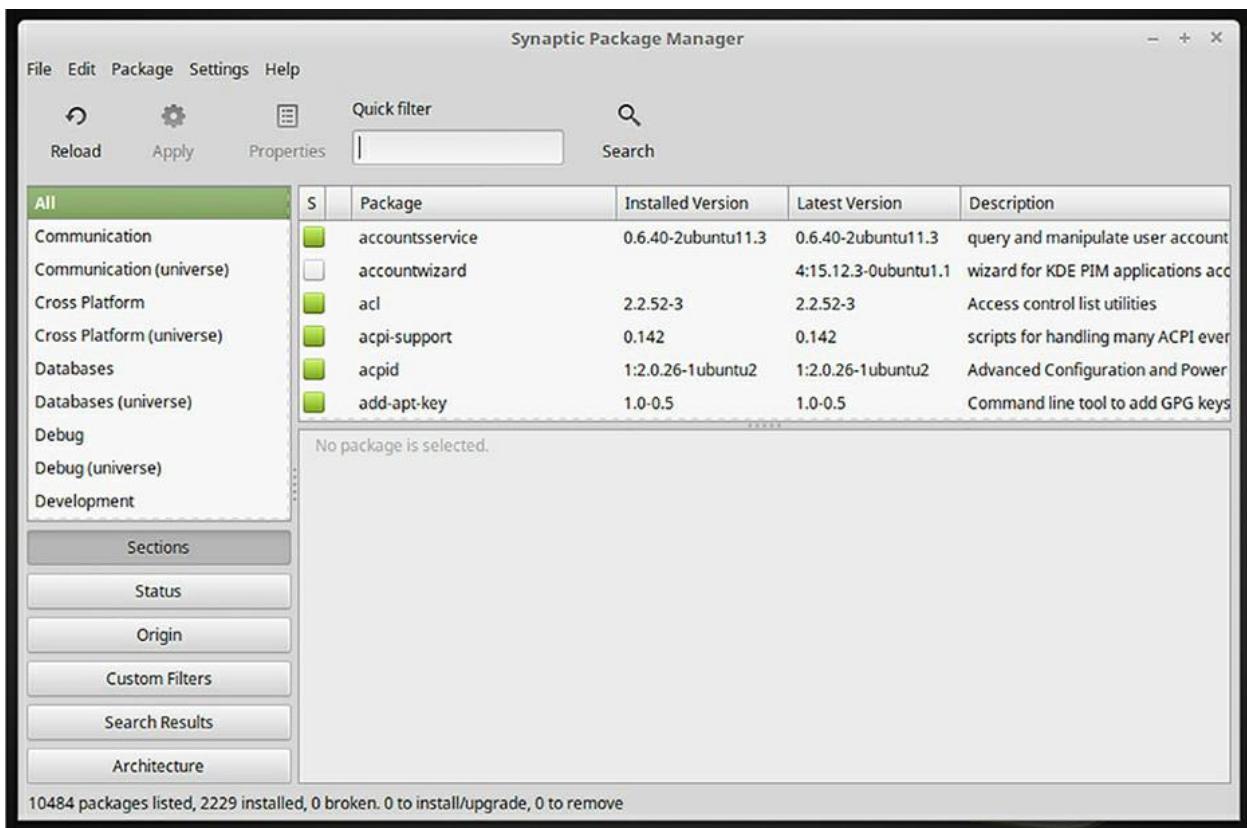


Figure 9.2 Synaptic enables you to search for, select, install, and uninstall software on Debian-based systems.

فایل‌های بسته Debian دارای نام‌هایی هستند که به **.deb** ختم می‌شوند. مانند بسته‌های RPM، این نام‌ها معمولاً شامل کدهایی برای نسخه نرمافزار و معماری (مانند 386 یا 64amd) هستند. به عنوان مثال، **samba\_3.6.1-3\_amd64.deb** یک فایل بسته **samba** برای بسته Debian، نسخه 3.6.1، ویرایش 3، برای پردازنده‌های AMD (64x86-64) است. شما می‌توانید چنین فایل‌هایی را با استفاده از **apt-get** یا **dpkg** نصب کنید، یا می‌توانید از **apt-get** برای دانلود یک بسته وابستگی‌های آن از اینترنت، با استفاده از دستور **install**، استفاده کنید، به صورت زیر:

```
# apt install samba
```

همانند بسته‌های RPM، شما می‌توانید بسته‌ها را حذف کنید یا نرم افزار کامپیوتر خود را ارتقاء دهید.

```
# apt remove zsh
```

```
# apt upgrade
```

APT یک ابزار قدرتمند است، همان‌طور که **dpkg** زیرساخت آن نیز همین‌گونه است. شما باید صفحات **man** این برنامه‌ها را مطالعه کنید تا اطلاعات بیشتری در مورد نحوه استفاده از این برنامه‌ها کسب کنید.

## Understanding the Process Hierarchy

کرنل لینوکس بخش مرکزی یک لینوکس است. کرنل Memory را مدیریت می‌کند، به نرم‌افزارها راهی برای دسترسی به هارد دیسک فراهم می‌کند، زمان CPU را تخصیص می‌دهد و وظایف حیاتی دیگر در سطح پایین را انجام می‌دهد. کرنل در اوایل فرایند بوت بارگذاری می‌شود و این کرنل است که مسئول مدیریت هر قطعه نرم‌افزاری دیگر روی یک کامپیوتر لینوکسی در حال اجرا است.

یکی از راههایی که کرنل برای اعمال نظم بر مجموعه‌ای از نرم‌افزارهای در حال اجرا که ممکن است آشفته باشند، ایجاد می‌کند، ایجاد نوعی سلسله‌مراتب است. هنگامی که بوت می‌شود، هسته فقط یک برنامه را اجرا می‌کند—معمولًاً یا `/sbin/init` / `/lib/systemd` یا `/init` راهاندازی تمامی برنامه‌های اساسی دیگری هستند که لینوکس باید اجرا کند، مانند برنامه‌هایی که مدیریت لاغین‌ها و سرورهای همیشه در حال اجرا را بر عهده دارند. چنین برنامه‌هایی، اگر مستقیماً توسط `systemd` یا `init` راهاندازی شوند، فرزندان (children) آن نامیده می‌شوند. فرآیندهای فرزند می‌توانند به نوبه خود فرزندان خود را راهاندازی کنند. این اتفاق زمانی رخ می‌دهد که شما وارد لینوکس می‌شوید. فرآیندی که یک فرآیند معین را راهاندازی کرده است، والد آن نامیده می‌شود.

نکته: شما می‌توانید با اضافه کردن گزینه `=init` به خط گزینه کرنل در بوت لودر خود، برنامه‌ای که به عنوان اولین فرآیند اجرا می‌شود را تغییر دهید، مانند `init=/bin/bash` برای اجرای `.bash`.

نتیجه این سیستم، سلسله‌مراتبی درختگونه از فرآیندها است، همان‌طور که در شکل 9.3 نشان داده شده است. ("Trees" در علم کامپیوتر اغلب به صورت وارونه نمایش داده می‌شوند). شکل 9.3 یک زیرمجموعه کوچک از بسیاری از فرآیندهایی که در یک لینوکس معمولی اجرا می‌شوند را نشان می‌دهد: فقط چند فرآیند مرتبط با لاغین تکست مود، شامل ابزار `login` که مدیریت لاغین‌ها را انجام می‌دهد، چند شل `bash` و چند برنامه کاربر. یک سیستم لینوکس کارا احتمالاً دهها یا صدها فرآیند در حال اجرا خواهد داشت. سیستمی که من در حال تایپ این کلمات روی آن هستم، 213 فرآیند را به طور همزمان اجرا می‌کند!

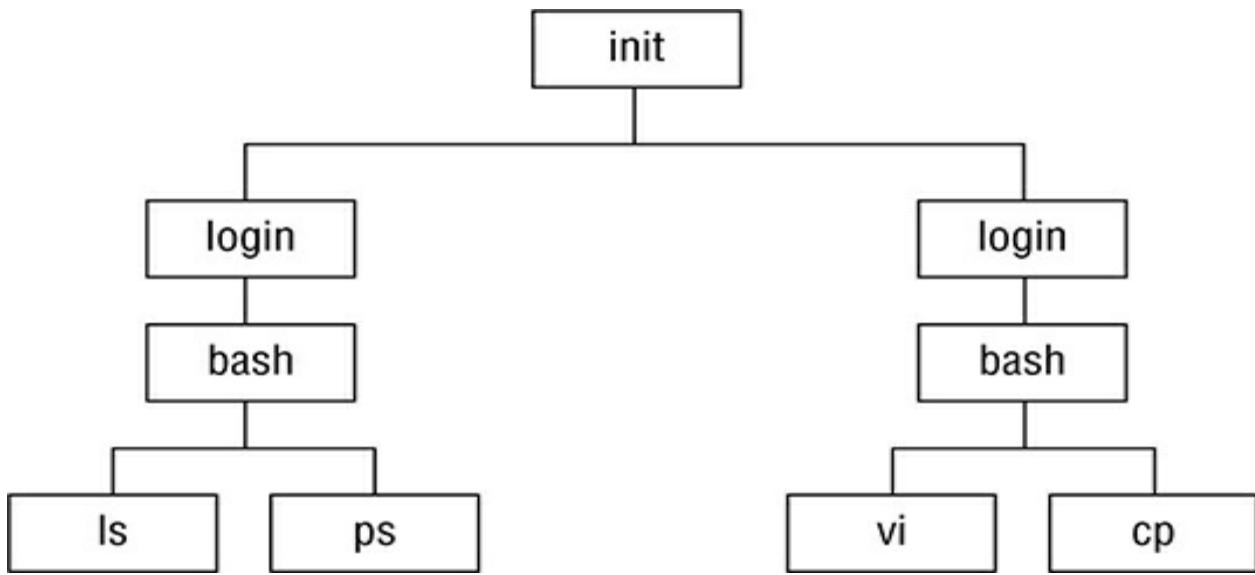


Figure 9.3 Linux processes are arranged in a hierarchical tree.

نکته: گاهی اوقات، یک فرآیند خاتمه می‌یابد اما فرزندان خود را پشت سر می‌گذارد. در این صورت، آن فرآیندهای فرزند را "adopt" می‌کند.

هر فرآیند دارای یک شماره شناسایی فرآیند (PID) مرتبط با آن است. این شماره‌ها از 1 شروع می‌شوند، بنابراین PID فرآیند init معمولاً 1 است. هر فرآیند همچنین یک شماره شناسایی فرآیند والد (PPID) دارد که به والد آن اشاره می‌کند. بسیاری از ابزارهای مدیریت فرآیندها به این شماره‌ها، به ویژه شماره PID، متکی هستند.

نکته: به طور داخلی، کرنل اطلاعات فرآیندها را در جدول فرآیند نگهداری می‌کند. ابزارهایی مانند ps و top (که به زودی توضیح داده می‌شوند) به شما امکان مشاهده و دستکاری این جدول را می‌دهند.

## Identifying Running Processes

قبل از اینکه بتوانید Process‌ها را مدیریت کنید، باید قادر به شناسایی آنها باشید. ابزارهای ps و top می‌توانند به شما در شناسایی فرآیندها کمک کنند. در هر دو حالت، می‌توانید فرآیندها را به روش‌های مختلف، مانند بر اساس نام یا استفاده از منابع، جستجو کنید. همچنین ممکن است بخواهید میزان حافظه‌ای که فرآیندهای شما مصرف می‌کنند را شناسایی کنید، که می‌توانید این کار را با دستور free انجام دهید.

## Using ps to Identify Processes

ابزار ساده‌تر برای شناسایی فرآیندها ps است که یک فهرست فرآیند تولید می‌کند. لیست 9.1 نمونه‌ای از عملکرد ps را نشان می‌دهد. در این مثال، گزینه لا-خروجی را به فرآیندهای متعلق به کاربر مشخص شده (rich) محدود می‌کند، در حالی که روابط والد/فرزنده را نشان می‌دهد.

Listing 9.1: Output of ps -u rich --forest

```
$ ps -u rich --forest
```

PID	TTY	TIME	CMD
2451	pts/3	00:00:00	bash
2551	pts/3	00:00:00	ps
2496	?	00:00:00	kvt
2498	pts/1	00:00:00	bash
2505	pts/1	00:00:00	\_ nedit
2506	?	00:00:00	\_ csh
2544	?	00:00:00	\_ xeyes
19221	?	00:00:01	dfm

لیست 9.2 یک مثال دیگر از دستور ps را نشان می‌دهد. در این مثال، گزینه u ستون‌های اطلاعاتی را اضافه می‌کند، در حالی که U خروجی را به فرآیندهای متعلق به rich محدود می‌کند. دستور ps از تعداد زیادی گزینه پشتیبانی می‌کند (برای جزئیات به صفحه man آن مراجعه کنید).

نکته: نسخه ps استفاده شده در اکثر توزیع‌های لینوکس ویژگی‌هایی از چندین پیاده‌سازی قبلی ps را ترکیب می‌کند. نتیجه این ترکیب، انتخاب بسیار بزرگی از گاهی اوقات گزینه‌های تکراری است.

Listing 9.2: Output of ps u U rich

```
$ ps u U rich
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
rich	19221	0.0	1.5	4484	1984	?	S	Sep07	0:01	dfm
rich	2451	0.0	0.8	1856	1048	pts/3	S	16:13	0:00	-bash
rich /opt/kd	2496	0.2	3.2	6232	4124	?	S	16:17	0:00	
rich	2498	0.0	0.8	1860	1044	pts/1	S	16:17	0:00	bash
rich	2505	0.1	2.6	4784	3332	pts/1	S	16:17	0:00	nedit
rich /bin/cs	2506	0.0	0.7	2124	1012	?	S	16:17	0:00	

rich	2544	0.0	1.0	2576	1360	?	S	16:17	0:00
xeyes									
rich	2556	0.0	0.7	2588	916	pts/3	R	16:18	0:00
U									

با توجه به تعداد زیاد گزینه‌های ps، کاربران مختلف ممکن است روش‌های مورد علاقه خود را برای استفاده از این برنامه داشته باشند. یک ترکیب محبوب از گزینه‌ها ax است که اطلاعاتی را که بیشتر مدیران سیستم نیاز دارند، ایجاد می‌کند، از جمله مقادیر PID و نام فرمان‌ها (شامل گزینه‌های کامند لاین) برای تمامی فرآیندها در کامپیوتر. اضافه کردن u (مانند aux) نام کاربری‌ها، بار CPU و چند تا دیگر اطلاعات را اضافه می‌کند. با این حال، حجم کلی اطلاعات تولید شده ممکن است زیاد باشد. یک راه برای محدود کردن این حجم، این است که نتایج را از طریق grep پرتاب کنید، که خطوطی را که شامل معیار جستجوی مشخص شده شما نیستند حذف می‌کند. به عنوان مثال، اگر می‌خواهید شماره PID فرآیند gedit را بدانید، می‌توانید این کار را به این شکل انجام دهید:

```
$ ps ax | grep gedit
27946 pts/8 Sl 0:00 gedit
27950 pts/8 S+ 0:00 grep --colour=auto gedit
```

نکته: به دلیل اینکه ps ax دستورات را به همراه گزینه‌هایش ایجاد می‌کند، استفاده از grep برای جستجوی یک رشته در خروجی، همچنین دستور grep خود را نیز برمی‌گرداند.

این دستور نشان می‌دهد که gedit دارای مقدار PID 27946 است. این معمولاً مهمترین اطلاعات هنگام استفاده از ps است، زیرا از مقدار PID برای تغییر اولویت یا پایان دادن به یک فرآیند استفاده می‌کنید.

## Using top to Identify Processes

اگرچه ps می‌تواند اطلاعات اولویت فرآیند و استفاده از CPU را بازگرداند، اما خروجی این برنامه معمولاً بر اساس شماره PID مرتب شده و فقط اطلاعات را در یک لحظه زمانی ارائه می‌دهد. اگر می‌خواهید به سرعت فرآیندهایی که مصرف حافظه یا CPU زیادی دارند را پیدا کنید، یا اگر می‌خواهید مطالعه کنید که چگونه استفاده از منابع در طول زمان تغییر می‌کند، ابزار دیگری مناسب‌تر است: top. این برنامه اصولاً یک نسخه تعاملی از ps است. شکل 9.4 نمایشی از top در یک پنجره ترمینال GNOME را نشان می‌دهد.

```

top - 10:12:52 up 6 min,  1 user,  load average: 1.16, 0.79, 0.42
Tasks: 181 total,   2 running, 178 sleeping,   0 stopped,   1 zombie
%Cpu(s):  1.7 us,  1.0 sy, 48.3 ni, 46.4 id,  2.6 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 5113276 total, 3787492 free, 668952 used, 656832 buff/cache
KiB Swap: 5280764 total, 5280764 free,     0 used. 4164088 avail Mem

PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM TIME+ COMMAND
3243 root      30  10 156368 96564 11668 R 99.0  1.9  0:27.35 update-apt-xapi
2178 rich      20   0 1821580 204892 80660 S  1.3  4.0  1:22.83 cinnamon
1026 root      20   0 358296 71516 38184 S  0.7  1.4  0:11.29 Xorg
3259 rich      20   0 41800  3644  3052 R  0.3  0.1  0:00.08 top
  1 root      20   0 119928  6004  3904 S  0.0  0.1  0:02.14 systemd
  2 root      20   0      0      0      0 S  0.0  0.0  0:00.00 kthreadd
  3 root      20   0      0      0      0 S  0.0  0.0  0:00.00 kworker/0:0
  4 root      0 -20      0      0      0 S  0.0  0.0  0:00.00 kworker/0:0H
  6 root      20   0      0      0      0 S  0.0  0.0  0:00.08 ksoftirqd/0
  7 root      20   0      0      0      0 S  0.0  0.0  0:00.44 rCU sched
  8 root      20   0      0      0      0 S  0.0  0.0  0:00.00 rCU_bh
  9 root      rt  0      0      0      0 S  0.0  0.0  0:00.00 migration/0
 10 root     0 -20      0      0      0 S  0.0  0.0  0:00.00 lru-add-drain
 11 root      rt  0      0      0      0 S  0.0  0.0  0:00.00 watchdog/0
 12 root     20   0      0      0      0 S  0.0  0.0  0:00.00 cpuhp/0
 13 root     20   0      0      0      0 S  0.0  0.0  0:00.00 cpuhp/1
 14 root      rt  0      0      0      0 S  0.0  0.0  0:00.00 watchdog/1
 15 root      rt  0      0      0      0 S  0.0  0.0  0:00.00 migration/1
 16 root     20   0      0      0      0 S  0.0  0.0  0:00.11 ksoftirqd/1
 18 root     0 -20      0      0      0 S  0.0  0.0  0:00.00 kworker/1:0H
 19 root     20   0      0      0      0 S  0.0  0.0  0:00.00 kdevtmpfs
 20 root     0 -20      0      0      0 S  0.0  0.0  0:00.00 netns
 21 root     20   0      0      0      0 S  0.0  0.0  0:00.00 khungtaskd
 22 root     20   0      0      0      0 S  0.0  0.0  0:00.00 oom_reaper
 23 root     0 -20      0      0      0 S  0.0  0.0  0:00.00 writeback
 24 root     20   0      0      0      0 S  0.0  0.0  0:00.00 kcompactd0

```

Figure 9.4 The top command shows system summary information and information about the most CPU-intensive processes on a computer.

به طور پیشفرض، top ورودی‌های خود را بر اساس استفاده از CPU مرتب می‌کند و نمایش خود را هر چند ثانیه به روز می‌کند. برای تشخیص اینکه یک برنامه که مصرف CPU بسیار بالایی دارد و رفتار نامطلوبی دارد یا نه، باید با اهداف و عادت‌های معمول برنامه‌های در حال اجرای سیستم خود آشنا باشید. در مثالی که در شکل 9.4 نشان داده شده است، یک فرآیند بهروزرسانی APT تقریباً تمام زمان CPU را اشغال کرده است. برای تعیین این موضوع، ورودی ستون %CPU را مشاهده کنید، سپس به سمت راست به ستون COMMAND بروید تا فرآیندی که زمان CPU را مصرف می‌کند را مشخص کنید. اگرچه این ابزار مفید است، اما مراقب باشید که به سرعت به استنتاجات برسید. در این مثال، بله، فرآیند بهروزرسانی APT در آن لحظه بسیار زمان CPU را مصرف می‌کند، اما این یک فعالیت کوتاه و چندین می‌ثبت زمانی است و پس از اتمام بهروزرسانی کاهش می‌یابد. نیازهای برنامه‌های مختلف به گونه‌ای متفاوت است که امکان ارائه یک قاعده ساده برای ارزیابی زمان CPU یک فرآیند به مقدار زیادی وجود ندارد.

شما می‌توانید با top بیشتر از تعماشای بهروزرسانی نمایشگر آن انجام دهید. وقتی اجرا می‌شود، می‌توانید هر یک از چندین دستور یک حرفی را وارد کنید، برخی از این دستورها شما را برای اطلاعات اضافی مورد نیاز فراخوانی می‌کنند، همانطور که در جدول 9.1 خلاصه شده است.

Table 9.1 Common top commands

Command	Description
h or ?	نمایش اطلاعات راهنما.
k	کشتن (کنسل) یک فرآیند. برنامه top از شما شماره PID را خواهد پرسید و در صورت امکان کشتن فرآیند، این کار را انجام می‌دهد.
q	خروج از top.
r	تغییر اولویت یک فرآیند.
s	نرخ بهروزرسانی نمایش را تغییر می‌دهد، که سپس شما آن را به ثانیه وارد می‌کنید.
P	تنظیم Display برای مرتب‌سازی بر اساس استفاده از CPU، که پیش‌فرض است.
M	تغییر Display به مرتب‌سازی بر اساس استفاده از حافظه.

یکی از اطلاعات ارائه شده توسط top، بار میانگین است، که یک اندازه‌گیری از تقاضای زمان پردازنده توسط برنامه‌ها می‌باشد. در شکل 9.4، شما می‌توانید سه براورد از میانگین بار را در خط بالایی ببینید؛ این‌ها مربوط به میانگین بار فعلی و دو اندازه‌گیری قبلی هستند. میانگین بار می‌تواند به شرح زیر تفسیر شود:

- سیستمی که هیچ برنامه‌ای تقاضای زمان پردازنده ندارد، میانگین بار آن 0 است.
- سیستمی که یک برنامه عملیاتی نیازمند به پردازنده اجرا می‌کند، میانگین بار آن 1 است.
- بارهای بالاتر در سیستم‌های تک‌پردازنده نشان‌دهنده برنامه‌هایی هستند که برای CPU Time موجود رقابت می‌کنند.

در یک کامپیوتر با چندین پردازنده یا هسته‌های پردازنده، میانگین بارها می‌تواند تا تعداد پردازنده‌ها یا هسته‌ها برسد قبل از اینکه رقابت برای زمان پردازنده آغاز شود. به عنوان مثال، میانگین بار 4.0 در سیستمی با پردازنده چهار هسته‌ای نشان‌دهنده فرآیندهایی است که دقیقاً به همان اندازه زمان پردازنده که کامپیوتر در دسترس دارد، نیاز دارند.

نکته: بیشتر کامپیوترهایی که امروز به فروش می‌رسند مدل‌های چند هسته‌ای هستند، اما اگر از لینوکس روی یک سیستم قدیمی‌تر استفاده می‌کنید، باید بدانید که مدل‌های تک‌هسته‌ای تا حدود سال 2006 در بازار غالب بودند. برای دیدن تعداد پردازنده‌های موجود در سیستم خود، از فرمان `lscpu` استفاده کنید.

متوسط بار می‌تواند در تشخیص پردازش‌های فراری مفید باشد. به عنوان مثال، اگر سیستمی به طور معمول دارای متوسط بار 0.5 باشد اما ناگهان به طور مداوم به متوسط بار 2.5 برسد، ممکن است چند فرآیند که منابع CPU را به شدت مصرف می‌کنند، دچار مشکل شده و به اصطلاح "آویزان" شده باشند - یعنی واکنش‌پذیر نباشند. فرآیندهای آویزان گاهی اوقات بدون نیاز مقدار زیادی از زمان CPU را مصرف می‌کنند. شما می‌توانید از `top` برای پیدا کردن این فرآیندها و در صورت لزوم متوقف کردن آن‌ها استفاده کنید.

نکته: دستور `w` که در فصل 13 توضیح داده شده است، می‌تواند به شما بگوید که Session های ترمینال چقدر زمان CPU را مصرف می‌کنند.

## Measuring Memory Use

فرآیندها تعدادی از منابع سیستم را مصرف می‌کنند که مهم‌ترین آن‌ها زمان CPU و RAM است. همانطور که قبلًا ذکر شد، دستور `top` به طور پیش‌فرض فرآیندها را بر اساس زمان CPU مرتب می‌کند تا بتوانید فرآیندهایی را که بیشترین زمان CPU را مصرف می‌کنند شناسایی کنید. می‌توانید با فشار دادن کلید `M` در داخل `top`، آن را به گونه‌ای تنظیم کنید که بر اساس استفاده از حافظه مرتب شود و فرآیندهایی که بیشترین حافظه را مصرف می‌کنند را شناسایی کنید. مانند زمان CPU، نمی‌توان گفت که یک فرآیند به دلیل قرار گرفتن در بالای لیست، بیش از حد حافظه مصرف می‌کند؛ برخی برنامه‌ها به طور مشروح مقدار زیادی حافظه مصرف می‌کنند. با این حال، گاهی اوقات یک برنامه به دلیل کدنویسی ناکارآمد یا نشت حافظه، بیش از حد حافظه مصرف می‌کند. نشت حافظه یک نوع باگ برنامه است که در آن برنامه از کرنل درخواست حافظه می‌کند و سپس پس از اتمام کار با حافظه، آن را بازنمی‌گرداند. یک برنامه با نشت حافظه مقدار بیشتری از حافظه را مصرف می‌کند، گاهی تا جایی که در عملکرد برنامه‌های دیگر اختلال ایجاد می‌کند. به عنوان یک راه حل کوتاه‌مدت، معمولاً می‌توانید برنامه را خاتمه داده و دوباره آن را اجرا کنید که مصرف حافظه برنامه را بازنشانی می‌کند، شبیه به تخلیه یک سینک که از یک شیر آبی که نشت دارد پر شده است، مشکل دوباره رخواهد داد، اما اگر نشت حافظه کوچک باشد، حداقل در این بین می‌توانید کار مفیدی انجام دهید.

نکته: کرنل به برنامه‌ها دسترسی به مجموعه‌ای از آدرس‌های حافظه را اعطای می‌کند که برنامه‌ها می‌توانند از آن‌ها استفاده کنند. هنگامی که یک برنامه کارش تمام شد، باید حافظه خود را به کرنل بازگرداند.

اگر می‌خواهید میزان استفاده کلی از حافظه کامپیوتر را بررسی کنید، دستور `free` مفید است. این برنامه گزارشی از وضعیت کلی حافظه کامپیوتر تولید می‌کند:

```
$ free
```

	total	used	free	shared	buffers	cached
Mem:	7914888	7734456	180432	0	190656	3244720
-/+ buffers/cache:		4299080	3615808			
Swap:	6291452	1030736	5260716			

خط Mem آمار حافظه (RAM) کلی را نشان می‌دهد، از جمله حافظه کل در کامپیوتر (بدون مقداری که توسط مادربرد و کرنل استفاده می‌شود)، میزان حافظه استفاده شده و میزان حافظه آزاد. در این مثال نشان داده شده است که بیشتر حافظه کامپیوتر در حال استفاده است. این وضعیت طبیعی است، زیرا لینوکس حافظه‌های غیر استفاده شده را به عنوان بافرها و حافظه‌های نهان استفاده می‌کند که به سرعت دسترسی به دیسک کمک می‌کند. بنابراین، خط `-/+ buffers/cache` این خط مجموع حافظه استفاده شده توسط برنامه‌های کامپیوتر را نشان می‌دهد. در این مثال، از مجموع 7,914,888 کیبی بایت، 4,299,080 در

حال استفاده است و 3,615,808 KiB آزاد است. به عبارت دیگر، بیش از نصف حافظه کامپیوتر توسط برنامه‌ها استفاده می‌شود، بنابراین مشکلات عملکردی مربوط به حافظه وجود ندارد.

خط Swap نشان می‌دهد چقدر فضای تعویض لینوکس در حال استفاده است. فضای تعویض فضای دیسک است که به عنوان یک افزونه به حافظه تعیین شده است. لینوکس زمانی از فضای تعویض استفاده می‌کند که حافظه RAM به پایان می‌رسد، یا زمانی که تصمیم می‌گیرد بهتر است که حافظه RAM برای بافرها یا حافظه‌های cache استفاده شود تا برای نگه داشتن برنامه‌های در حال حاضر غیرفعال. در این مثال، از مجموع 6,291,452 کیبی بایت، 1,030,736 کیبی بایت فضای تعویض استفاده شده است و 5,260,716 KiB آزاد است. معمولاً استفاده از فضای تعویض بسیار کم است، و اگر بسیار زیاد بشود، ممکن است با مشکلات عملکردی مواجه شوید. در طولانی‌مدت، افزایش حافظه RAM کارآمدترین راه حل برای این مشکلات است. اگر به دلیل استفاده بیش از حد از فضای تعویض از مشکلات عملکردی رنج می‌برید و نیاز به کمک فوری دارید، متوقف کردن برنامه‌هایی که حافظه زیادی مصرف می‌کنند، می‌تواند کمک کند. نشت حافظه، که قبلاً شرح داده شد، می‌تواند منجر به این مشکلات شود، و متوقف کردن برنامه‌ای که دارای نشت حافظه است، می‌تواند عملکرد سیستم را به حالت عادی بازگرداند. دستور free از تعدادی گزینه پشتیبانی می‌کند، اکثر این‌ها فرمت نمایش آن را تغییر می‌دهند. مفیدترین این‌ها m-است که باعث استفاده از واحدهای مبی بایت (MiB) به جای واحدهای پیش‌فرض کیبی بایت (KiB) می‌شود.

## Using Log Files

برخی از برنامه‌هایی که به صورت بک گراند (یا به عبارت دقیق‌تر دیمون) اجرا می‌شوند، اطلاعات مربوط به عملکرد عادی خود را به فایل‌های log می‌نویسند، که فایل‌هایی هستند که چنین یادداشت‌هایی را ثبت می‌کنند. مشاوره با فایل‌های log می‌تواند بخش مهمی از تشخیص مشکلات با دیمون‌ها باشد. اولین گام در انجام این کار، پیدا کردن فایل‌های log شما است. در برخی موارد، ممکن است نیاز باشد که برنامه، برای کمک به پیدا کردن مشکل، خروجی بیشتری را تولید کند، بنابراین این بخش برخی راهنمایی‌ها در این زمینه ارائه می‌دهد. این بخش همچنین درباره حلقه حافظه کرنل که به طور فنی یک فایل log نیست اما می‌تواند نقش مشابهی را برای اطلاعات کرنل داشته باشد، توضیح می‌دهد.

## Locating Log Files

لينوکس اکثر فایل‌های log را در درخت دایرکتوری /var/log / ذخیره می‌کند. برخی از فایل‌های log در همان دایرکتوری قرار دارند، اما برخی از سرورها زیردایرکتوری‌های کاملی را برای ذخیره فایل‌های log خود ایجاد می‌کنند. جدول 9.2 خلاصه‌ای از برخی از فایل‌های log را رایج در بسیاری از سیستم‌های لینوکس است. علاوه بر این، بسیاری از برنامه‌های سرور که در این کتاب توضیح داده نشده‌اند، فایل‌های log یا زیردایرکتوری‌های خود را در /var/log / اضافه می‌کنند. اگر با چنین سروری مشکل دارید، بررسی فایل‌های log آن می‌تواند مکان خوبی برای شروع رفع مشکل باشد.

نکته: جزئیات فایل‌های log بین توزیع‌های لینوکس متفاوت است، بنابراین برخی از فایل‌های موجود در جدول 9.2 ممکن است در سیستم شما وجود نداشته باشند یا فایل‌هایی که پیدا می‌کنید نام‌های متفاوتی داشته باشند.

Table 9.2 Important log files

Log file	Contents
boot.log	سرویس‌هایی که به وسیله اسکریپت‌های شروع SysV در فرآیند بوت بعداً شروع می‌شوند.
cron	فرآیندهایی که به وسیله دیمون cron در فواصل زمانی منظم اجرا می‌شوند. اگرچه این کتاب درباره cron توضیح نمی‌دهد، اما یک مشکل با آن ممکن است باعث ایجاد مشکلاتی شود که به طور منظم تکرار می‌شوند، بنابراین شما باید از آن اطلاع داشته باشید.
cups/	دایرکتوری حاوی فایل‌های گزارش‌های مربوط به سیستم پرینت لینوکس
gdm/	دایرکتوری حاوی فایل‌های گزارش‌های مربوط به مدیریت Display گنوم (GDM)، که ورودی‌های گرافیکی را در بسیاری از سیستم‌ها مدیریت می‌کند
messages or syslog	یک فایل گزارش عمومی که شامل پیام‌هایی از بسیاری از دیمون‌ها است که از فایل‌های گزارش اختصاصی خود برخوردار نیستند
secure	پیام‌های مربوط به امنیت، شامل اعلامیه‌هایی از زمانی که کاربران از ابزارهایی مانند su، sudo، و ابزارهای مشابه برای کسب امتیازهای روت استفاده می‌کنند
Xorg.0.log	اطلاعات در مورد آخرین شروع مجدد سیستم X

فایل‌های لاغ زیادی به دوره می‌انجامند، به این معنی که قدیمی‌ترین فایل لاغ حذف می‌شود، جدیدترین فایل لاغ به یک تاریخ یا شماره تغییر نام می‌دهد، و یک فایل لاغ جدید ایجاد می‌شود. به عنوان مثال، اگر در 1 دسامبر 2019 مورد استفاده قرار گیرد، var/log/messages به var/log/messages-20191201، var/log/messages به var/log/messages-1.gz، یا موارد مشابه تغییر نام می‌دهد و یک var/log/messages-1.gz جدید ایجاد می‌شود. این روش از اینکه فایل‌های لاغ بیش از حد از کنترل خارج نشوند جلوگیری می‌کند.

نکته: چرخه‌ی گردش فایل‌های لاغ در آخرین ساعت شب رخ می‌دهد، بنابراین اگر کامپیوتر خود را خاموش کنید، این اتفاق نخواهد افتاد. برای اطمینان از چرخه‌ی گردش فایل‌های لاغ، به طور دوره‌ای کامپیوتر را شبها روشن نگه دارید.

بیشتر فایل‌های لاغ فایل‌های متن ساده هستند، بنابراین می‌توانید آن‌ها را با استفاده از هر ابزاری که قادر به بررسی فایل‌های تکست است، مانند less یا ویرایشگر متنی، بررسی کنید. یک دستور بسیار مفید استفاده از

دستور tail است که آخرین 10 خط یک فایل را نشان می‌دهد (یا تعداد خطوطی که با گزینه -n مشخص می‌کنید). به عنوان مثال، تایپ کردن tail /var/log/messages 10 خط این فایل را نشان می‌دهد.

توجه داشته باشید که همه برنامه‌ها پیام‌ها را ثبت نمی‌کنند. به طور معمول، فقط دیمون‌ها این کار را انجام می‌دهند؛ برنامه‌های کاربر عادی پیام‌های خطا را به روش‌های دیگر نمایش می‌دهند، در باکس‌های GUI دیالوگ یا در یک ترمینال تکست مود. اگر فکر می‌کنید که یک برنامه باید داده‌ها را به ثبت برساند اما نمی‌توانید آن را پیدا کنید، به دلایلی که آن مراجعه کنید. به طور جایگزین، می‌توانید از دستور grep برای تلاش برای پیدا کردن فایل log که برنامه داده‌های خود را به آن ارسال می‌کند، استفاده کنید. به عنوان مثال، تایپ کردن grep sshd /var/log/\* (نام دیمون SSH) در آن ظاهر می‌شود، پیدا می‌کند.



## Real World Scenario

### Creating Log Files

برخی از برنامه‌ها فایل‌های log خود را ایجاد می‌کنند؛ با این حال، بیشتر آن‌ها بر روی یک ابزار معمولاً با نام جانشینی به نام دیمون لاغ سیستم (System Log Daemon) برای انجام این کار وابسته هستند. نام فرآیند این برنامه عموماً syslog یا journald است. مانند سایر دیمون‌ها، این توسط اسکریپت‌های راهاندازی سیستم در فرآیند بوت شروع می‌شود. چندین بسته دیمون لاغ سیستم در دسترس است. برخی از آن‌ها ابزار جداگانه‌ای مانند klogd یا klogd را ارائه می‌دهند تا پیام‌های لاغ از کرنل را به طور جداگانه از برنامه‌های عادی مدیریت کنند.

شما می‌توانید رفتار دیمون لاغ را تغییر دهید، از جمله تنظیم فایل‌هایی که پیام‌های خاصی را به آن‌ها لاغ می‌کند، با تنظیم فایل پیکربندی آن. نام این فایل به دیمون خاص مورد استفاده بستگی دارد، اما معمولاً /etc/rsyslog.conf یا چیزی مشابه آن است. جزئیات پیکربندی فایل لاغ خارج از محدوده این کتاب است، اما شما باید بدانید که چنین جزئیاتی می‌توانند تغییر کنند. این واقعیت دلیل بسیاری از تفاوت‌های ویژگی‌های فایل لاغ بین توزیع‌های مختلف است.

بعد از اجرا، یک دیمون لاغ پیام‌ها را از فرآیندهای دیگر با استفاده از تکنیکی به نام پیام‌رسانی سیستم (system messaging) می‌پذیرد. سپس پیام‌ها را مرتب می‌کند و بر اساس منبع پیام و یک کد اولویت، آن‌ها را به یک فایل لاغ مناسب هدایت می‌کند.

## Producing More Verbose Log File Entries

گاهی اوقات فایل‌های لاغ اطلاعات کافی برای شناسایی منبع مشکل ارائه نمی‌دهند. خوشبختانه، بسیاری از برنامه‌هایی که خروجی لاغ فایل تولید می‌کنند، می‌توانند طوری پیکربندی شوند که خروجی بیشتری تولید

کنند. متأسفانه، این کار گاهی اوقات می‌تواند باعث شود که جستجو در میان تمام ورودی‌ها برای یافتن اطلاعات مرتبط سخت‌تر شود.

روش افزایش جزئیات خروجی لاگ فایل‌ها از یک برنامه دیگر متفاوت است. معمولاً باید گزینه‌ای را در فایل پیکربندی برنامه تنظیم کنید. برای یادگیری چگونگی انجام این کار، باید به مستندات برنامه مراجعه کنید.

## Examining the Kernel Ring Buffer

بافر حلقه‌ای هسته شبیه به یک فایل لاگ برای هسته است؛ با این حال، برخلاف دیگر فایل‌های لاگ، در حافظه ذخیره می‌شود و نه در فایل دیسک. مشابه فایل‌های لاگ معمولی، محتوای آن با ادامه کار کامپیوتر تغییر می‌کند. برای بررسی بافر حلقه‌ای هسته، می‌توانید دستور `dmesg` را تایپ کنید. انجام این کار خروجی فراوانی ایجاد می‌کند، بنابراین معمولاً خروجی را از طریق `less` انتقال می‌دهید:

```
$ dmesg | less
```

نکته: از آنجا که بافر حلقه‌ای کرنل دارای اندازه محدودی است، ورودی‌های اولیه آن ممکن است در صورتی که کامپیوتر برای مدت طولانی کار کند یا اگر چیزی تعداد زیادی ورودی تولید کند، از دست بروند.

به طور متناوب، اگر بدانید که اطلاعاتی که می‌خواهید با یک رشته خاص مرتبط خواهد بود، می‌توانید از grep برای جستجوی آن استفاده کنید. به عنوان مثال، برای یافتن پیام‌های بافر حلقه‌ای هسته در مورد اولین دیسک سخت، `/dev/sda`، ممکن است دستور زیر را تایپ کنید:

```
$ dmesg | grep sda
```

پیام‌های بافر حلقه‌ای کرنل ممکن است بسیار مبهم باشند؛ اما می‌توانند در تشخیص مشکلات سخت‌افزاری و درایورها بسیار ارزشمند باشند، چرا که وظیفه کرنل، برقراری ارتباط با سخت‌افزار است. اگر یک دستگاه سخت‌افزاری رفتار عجیب دارد، ممکن است بخواهید بافر حلقه‌ای کرنل را جستجو کنید. حتی اگر پیامی که پیدا می‌کنید را درک نمی‌کنید، می‌توانید آن پیام را در یک موتور جستجوی وب جستجو کنید یا آن را به یک همکار با دانش بیشتر منتقل کنید تا راهنمایی بگیرید.

برخی توزیع‌ها یک نسخه از بافر حلقه‌ای کرنل را در هنگام اولین بوت سیستم در `/var/log/dmesg` یا فایل مشابه قرار می‌دهند. اگر کامپیوتر به قدری طولانی اجرا شده باشد که ورودی‌های اولیه آن از بین رفته باشد، می‌توانید این فایل را بررسی کنید. اگر می‌خواهید چنین فایلی را در توزیعی که به طور پیشفرض این کار را نمی‌کند ایجاد کنید، می‌توانید فایل `/etc/rc.d/rc.local` را ویرایش کرده و خط زیر را به انتهای آن اضافه کنید:

```
dmesg > /var/log/dmesg
```

# CHAPTER 10

## Editing Files

دکیومنت های کامپیوترا به اشکال مختلفی وجود دارند، اما یکی از پایه‌ای‌ترین و انعطاف‌پذیرترین‌ها فایل‌های متنی هستند. به طور معمول، فایل‌های پیکربندی و اسکریپت‌های شل فایل‌های متنی هستند. از آنجا که شما اغلب در حال تغییر فایل‌های پیکربندی و ایجاد اسکریپت‌های شل هستید، باید قادر به ویرایش فایل‌های متنی باشید. این فصل به این کار با تأکید بر ویرایشگرهای متنی ساده مدل nano و vi می‌پردازد. ابتدا چند نقش توسط فایل‌های متنی توضیح داده شده و سپس نحوه انتخاب ویرایشگر متنی توضیح داده شده است. برای ویرایش فایل‌های متنی، البته باید قادر به شروع ویرایشگر باشید، سپس یا بر روی یک سند موجود یا برای ایجاد یک سند جدید. ویرایشگر nano به طور کلی نسبتاً ساده است، بنابراین عملکرد آن ابتدا توضیح داده می‌شود، سپس ویرایشگر vi، که توسط استانداردهای مدرن کمی عجیب‌تر است، توضیح داده می‌شود.

### Understanding the Role of Text Files

یک ویرایشگر متنی به شما اجازه می‌دهد تا اسنادی را که در قالب متن ساده ذخیره شده‌اند، ویرایش کنید. کدهای استاندارد آمریکایی برای تبادل اطلاعات (ASCII) قبلًا شکل متداولی داشت، اما اکنون فایل‌ها معمولاً از فرمتهای یونیکد برای پشتیبانی از کاراکترهای اضافی استفاده می‌کنند.

نکته: فایل‌های متنی پایان خطوط را با استفاده از یک یا دو کاراکتر ویژه ASCII کدگذاری می‌کنند. رمزگذاری پایان خط میان سیستم عامل‌های یونیکس (یا لینوکس) و ویندوز متفاوت است، اما اکثر برنامه‌ها قادر به کار با هر دو روش هستند.

این فرمتهای اسناد متنی را ذخیره می‌کنند که به تنها یی شامل هیچ قالب‌بندی ویژه یا ویژگی‌های جاسازی شده‌ای نیستند. فایل‌های متنی نمی‌توانند شامل گرافیک، استفاده از فونت‌های مختلف، تاکید کلمات با ایتالیک کردن آن‌ها یا استفاده از سایر ویژگی‌هایی که شما احتمالاً با Word Processor ها مرتبط می‌کنید (اگرچه ابزارهای نشانه‌گذاری استثنایی جزئی را به این قاعده اضافه می‌کنند) شوند.



### Real World Scenario

ASCII and Unicode

استاندارد ASCII به دهه 1960 بر می‌گردد. این یک کد 7 بیتی است، به این معنا که حداقل ۲<sup>7</sup> یا 128 کاراکتر را پشتیبانی می‌کند. (در عمل، ASCII از 8 بیت استفاده می‌کند، بنابراین 128 کاراکتر اضافی در دسترس هستند. این بیت‌ها کاراکترهای کنترل مختلف را کد می‌کنند یا در توسعه‌های ASCII استفاده می‌کنند.)

می‌شوند). ASCII برای کد کردن حروف استفاده شده در انگلیسی، ارقام و نمادها ایجاد شد. این هدف اصلی، به همراه تعداد محدود کاراکترهای ASCII، آن را برای بسیاری از زبان‌های غیرانگلیسی مفید نمی‌سازد. ASCII به اندازه کافی کاراکتر برای مدیریت تمام نیازهای زبان‌های غیرانگلیسی ندارد.

طی سال‌ها، توسعه‌ها و نسخه‌های متفاوتی از استاندارد اسکی برای پشتیبانی از کاراکترها و الفباهای اضافی که استاندارد اسکی پشتیبانی نمی‌کند، استفاده شده است. یک روش برای این کار استفاده از یک صفحه کد برای مشخص کردن الفبا است. هر صفحه کد نوعی از استاندارد اسکی را مشخص می‌کند که برای یک الفبای خاص مناسب است. به عنوان مثال، صفحه کد 866 الفبای سیریلیک را (الفبای استفاده شده در زبان روسی و بیشتر زبان‌های اسلامی دیگر) کدگذاری می‌کند. مشکل با صفحه‌های کد این است که شما به طور کلی تنها می‌توانید یکی را در هر زمان استفاده کنید.

یونیکد رویکردی مدرن‌تر است. این استاندارد مجموعه کاراکترهای بسیار بزرگی را فراهم می‌کند که امکان رمزگذاری هر الفبایی که در استفاده روزمره جهان وجود دارد را فراهم می‌کند، از جمله سیستم‌های نوشتاری لوگو گرافیک بزرگ که در زبان‌هایی مانند چینی و ژاپنی استفاده می‌شوند. مشکل این است که یونیکد نیاز به تعدادی بیشتری بیت دارد، و چندین روش برای رمزگذاری بهینه آن وجود دارد. خوشبختانه، این روش‌های تبدیل فرمت یونیکد (UTF) نسبت به صفحه‌های کد به تعداد کمتری محدود است. برخی از این روش‌ها، مانند UTF-8، کاراکترهای اولیه را به همان روش استاندارد اسکی نگاشت می‌کنند، بنابراین یک فایل اسکی همچنین یک فایل UTF-8 معتبر است. بسیاری از ویرایشگرهای متن امروزی به طور خودکار با فرمت UTF-8 (یا سایر فرمت‌های یونیکد) سازگار هستند، بنابراین می‌توانید از یک ویرایشگر متن برای نوشتن فایل‌های متنی به هر زبانی که دوست دارید استفاده کنید. ممکن است هنوز نیاز به تنظیم گزینه‌های کاستوم داشته باشید تا لینوکس را مطلع کنید که از چه نوع صفحه‌کلیدی استفاده می‌کنید و چه صفحه کدی برای برنامه‌هایی که هنوز هم به صفحه‌های کد وابسته هستند، به صورت پیش‌فرض استفاده کند.

اگر فایل متنی شما با کدگذاری ASCII باشد، در واقع به صورت یونیکد هم رمزگذاری می‌شود. رمزگذاری ASCII به عنوان یک زیرمجموعه از یونیکد محسوب می‌شود.

فایل‌های متنی از خطوطی تشکیل شده‌اند که می‌توانند از 0 کاراکتر تا اندازه کل فایل متغیر باشند و هر نوع داده‌ای را نگه دارند. ممکن است بخواهید برخی از آن‌ها را به عنوان یک کاربر عادی ایجاد یا ویرایش کنید؛ بعضی از آن‌ها اهمیت برای مدیریت یک سیستم لینوکس دارند. انواع اصلی فایل عبارتند از:

- فایل‌های زبان انسان
- فایل‌های زبان برنامه‌نویسی
- فایل‌های متن قالب‌بندی‌شده
- فایل‌های کانفیگ برنامه و سیستم
- فایل‌های لاگ برنامه

نکته: فایل‌های متن قالب‌بندی شده حاوی قوانین و نحوه قالب‌بندی خاصی هستند که با استفاده از دنباله‌های کاراکترهای منحصر به فردی رمزگذاری شده‌اند. اگرچه شما می‌توانید این نوع فایل‌ها را با یک ویرایشگر متنی ویرایش کنید، اما ویرایشگرهای ویژه‌ای هم برای بسیاری از این انواع فایل وجود دارند.

برخی از فایل‌ها شامل عناصر چندین دسته هستند. به عنوان مثال، ایمیل می‌تواند در فایل‌های متنی ذخیره شود. یک فایل ایمیل به طور عمده شامل زبان انسانی است، اما پیام‌های ایمیل شامل هدرها نیز هستند که منبع و مقصد کامپیوترها را توصیف می‌کنند، همراه با اطلاعاتی درباره نحوه انتقال پیام از یک سایت به دیگری که به شکلی شبیه به داده‌های متن قالب‌بندی شده یا فایل‌های ورودی-خروجی هستند.

## Choosing an Editor

تمام توزیع‌های لینوکس با بسیاری از ویرایشگرهای متنی ارائه می‌شوند. به طور کلی، ویرایشگرهای متنی به یکی از دو دسته زیر تقسیم می‌شوند: ویرایشگرهای تکست مود و GUI. معمولاً مبتدیان با ویرایشگرهای گرافیکی راحت‌تر هستند که حتی برای متخصصان نیز می‌تواند راحت‌تر باشد. اما هنگامی که یک GUI در دسترس نیست، شما ممکن است مجبور شوید از یک ویرایشگر تکست مود استفاده کنید. بنابراین، بهتر است با حداقل یک ویرایشگر تکست مود آشنا شوید.

برخی از ادیتورهای تکست مود محبوب عبارتند از:

ویرایشگر vi یکی از استانداردهای یونیکس است. کوچک است و معمولاً به طور پیش‌فرض نصب می‌شود، بنابراین می‌توانید اطمینان حاصل کنید که در هر کامپیوتر لینوکسی وجود دارد. با این حال، با استانداردهای مدرن عجیب است - از چندین حالت ویرایش استفاده می‌کند و شما باید بین آنها سوئیچ کنید تا وظایف مختلفی را انجام دهید. بسیاری از ادمین‌های یونیکس و لینوکس با سابقه vi را به دلیل انعطاف‌پذیری، قدرت و اندازه کوچک آن دوست دارند.

نکته: بیشتر توزیع‌های لینوکس از یک نسخه از vi به نام "vi improved" یا vim استفاده می‌کنند؛ به طور معمول هنوز می‌توانید آن را با تایپ vi اجرا کنید.

ویرایشگر emacs یکی دیگر از ابزارهای معمول در یونیکس است. این یک ویرایشگر بزرگ با امکانات فراوان است، بنابراین کمتر احتمال دارد به صورت پیش‌فرض نصب شود، به ویژه در توزیع‌های کوچک و سبک. مدل عملکرد آن شبیه به ویرایشگرهای متدوال برای مبتدیان است، اما دستورات آن ممکن است کمی عجیب به نظر برسند.

نکته: دستورات ویرایش متن در Bash بر اساس دستورات emacs مدل‌سازی شده‌اند، بنابراین یادگیری می‌تواند توانایی شما در کار با شل Bash را بهبود بخشد.

**nano**: چندین ویرایشگر کوچک بر اساس emacs مدل‌سازی شده‌اند، اما بسیاری از ویژگی‌های پیشرفته آن را برای ساده‌تر کردن ویرایشگر حذف می‌کنند. یکی از این ویرایشگرها nano است که کوچک، سبک و آسان برای استفاده است.

ویرایشگر nano احتمالاً بهترین مکان برای شروع است، به دلیل سهولت استفاده‌اش. اگر nano از قبل نصب نشده باشد، معمولاً در ریپاپیتوری نرم‌افزاری بیشتر توزیع‌ها موجود است. (اگر nano را روی توزیع خود پیدا نکردید، به فصل 9، "Exploring Processes and Process Data" مراجعه کنید تا کمک برای نصب بسته nano بگیرید.) شکل 10.1 نشان می‌دهد که در یک session لایکن تکست مود در حال ویرایش فایلی به نام pets.txt است.

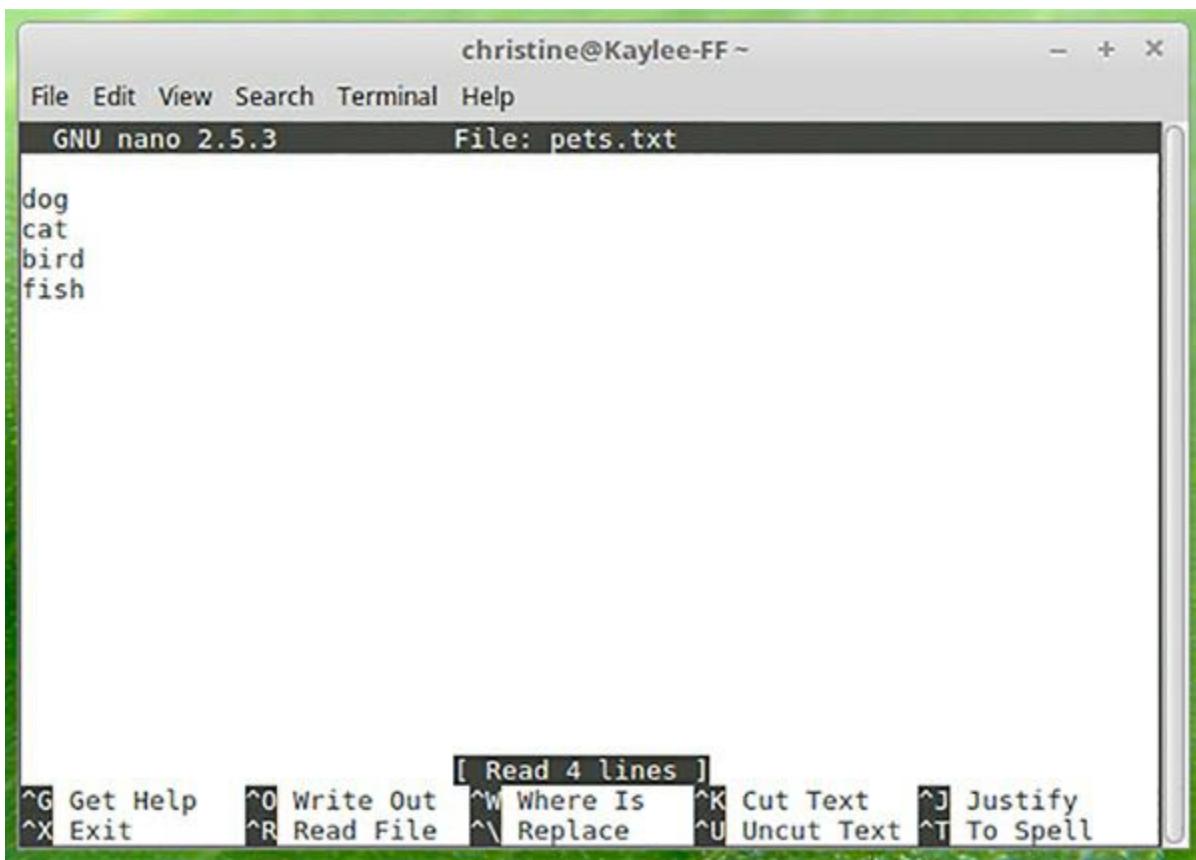


Figure 10.1 The nano editor enables you to edit a text file in text mode

مانند ویرایشگرهای متنی، چندین ویرایشگر GUI نیز در دسترس هستند، از جمله موارد زیر:

**emacs**: ویرایشگر emacs هم یک ویرایشگر متنی و هم یک ویرایشگر GUI است. با این حال، ویژگی‌های GUI در emacs گاهی اوقات کمی عجیب به نظر می‌رسند؛ به عنوان مثال، نوار اسکرول برای حرکت در فایل در سمت چپ پنجره ظاهر می‌شود نه در سمت رایج‌تر راست.

**gedit**: محیط دسکتاپ GNOME یک ویرایشگر متنی مرتبط به نام gedit دارد. این یک ویرایشگر متنی نسبتاً معمولی است و اغلب به صورت پیش‌فرض نصب می‌شود.

**Kate و KWrite**: همانطور که gedit با GNOME مرتبط است، Kate و KWrite ویرایشگرهای هستند که با محیط دسکتاپ KDE مرتبط هستند. KWrite کمی پیشرفته‌تر از gedit است و Kate برخی ویژگی‌های بیشتری اضافه می‌کند، اما هیچ‌کدام به اندازه emacs قدرتمند نیستند.

**Geany**: ویرایشگری است که به هیچ محیط دسکتاپ خاصی وابسته نیست و کوچک، سبک و نسبتاً قدرتمند است. همچنین تحت سیستم‌عامل‌های دیگر به جز لینوکس، مانند ویندوز نیز اجرا می‌شود که اگر بخواهید از یک ویرایشگر برای چندین پلتفرم استفاده کنید، بسیار مفید است.

برای یک کاربر جدید لینوکس، هر یک از این ویرایشگرهای شروع خوب برای ویرایشگر GUI هستند؛ همه آن‌ها ویژگی‌های اساسی که برای ویرایش سبک فایل‌های متند نیاز دارید را ارائه می‌دهند. انتخاب شما ممکن است به این بستگی داشته باشد که کدام یک به طور پیش‌فرض روی سیستم شما نصب شده است. در درازمدت، احتمالاً باید انواع مختلفی از ویرایشگرهای امتحان کنید تا ویرایشگری که بیشتر دوست دارید را پیدا کنید.

## Editing Files with nano

اگر با ویرایشگرهای متون تکست مود آشنا هستید، یادگیری nano برای شما مشکل چندانی نخواهد داشت. اگر تاکنون فقط از ویرایشگرهای GUI برای ویرایش متون استفاده کرده‌اید، باید چندین قرارداد کیبورد را یاد بگیرید. برای مثال، باید به جای استفاده از ماوس با استفاده از کیبورد در سند جایه‌جا شوید. می‌توانید همانند ویرایشگر متون یا پردازشگر کلمه GUI، متون را وارد، جایگزین و حذف کنید.

نکته: اگر nano بر روی سیستم شما نصب نشده است، می‌توانید از اطلاعات فصل 9 برای نصب بسته nano استفاده کنید.

شما می‌توانید ویرایشگر متون nano را از کامند لاین اجرا کنید. می‌توانید این کار را به عنوان یک کاربر عادی انجام دهید و با استفاده از مجوزهای پیش‌فرض خود، فایل‌های متونی که مالک آن‌ها هستید را ویرایش کنید. برای ویرایش فایل‌های سیستم، باید از مجوزهای کاربر ادمین (super user) استفاده کنید. در فصل 12، "Understanding Basic Security" ویرایش فایل‌های متون سیستم را بدست آورید. در این مثال، nano راهاندازی می‌شود:

```
$ nano
```

وقتی باز شود، یک نمایش مشابه با تصویر 10.1 را مشاهده خواهید کرد، اما بیشتر پنجره یا کنسول تکست مود خالی خواهد بود، زیرا شما نام فایل را مشخص نکرده‌اید. به جای اینکه مانند تصویر 10.2 نشان داده شود، وسط خط بالای New Buffer خواهد بود. شما می‌توانید شروع به تایپ متون مورد نظر خود کنید. وقتی فایل را ذخیره می‌کنید، همانطور که در "Saving Your Changes from nano" توضیح داده شده است، nano از شما برای نام فایل درخواست خواهد کرد.

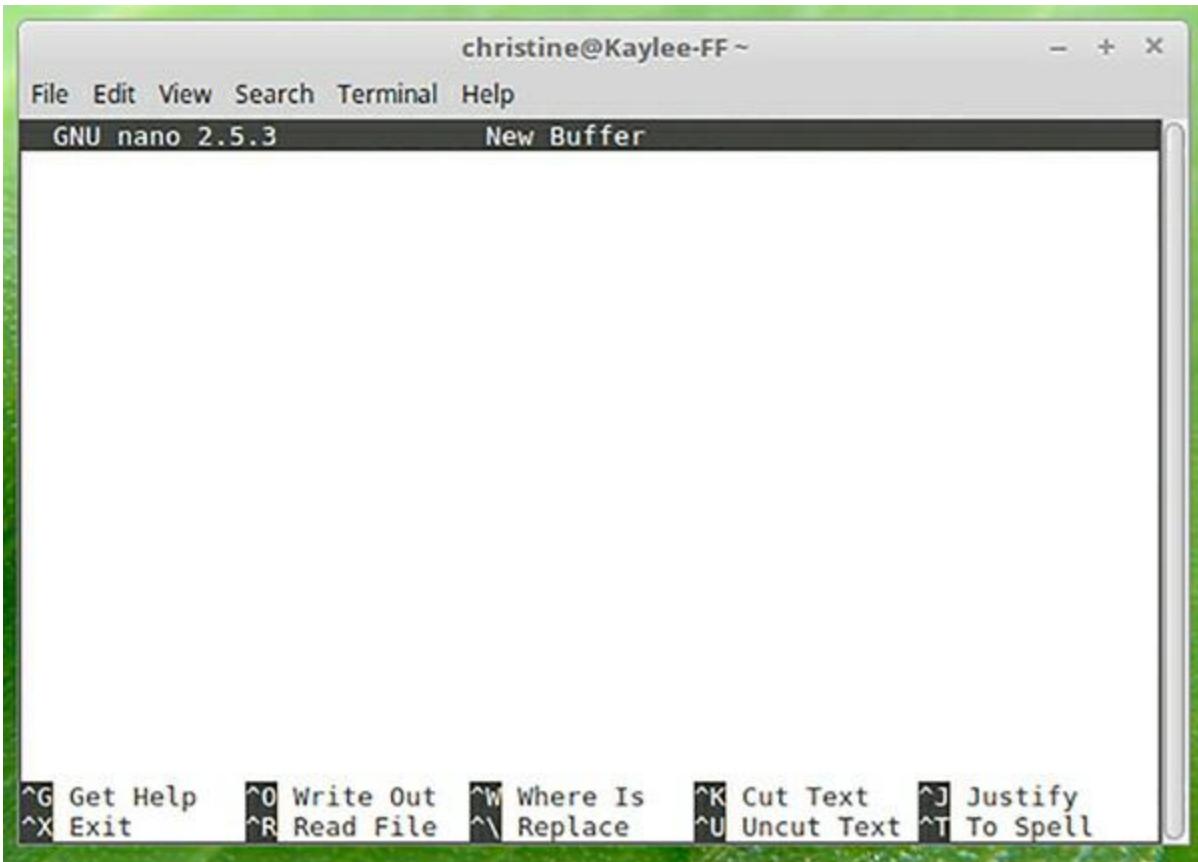


Figure 10.2 The nano editor launched with no filename provided.

به طور جایگزین، می‌توانید هنگام اجرای ادیتور متن یک نام فایل ارائه دهید به این صورت:

```
$ nano great_american_novel.txt
```

این مثال فایل great\_american\_novel.txt را باز کرده و نمایش می‌دهد. اگر فایل وجود نداشته باشد، nano پیغام "New File" را در نزدیکی پایین نمایشگر نمایش می‌دهد، جایی که در شکل 10.1 پیغام "Read 4 lines" دیده می‌شود. اگر در خط سوم از پایین پیغام "Warning: no write permission" را مشاهده کردید، یعنی فایل را بارگذاری کرده‌اید که مجوز تغییر آن را ندارید. برای ذخیره تغییرات فایل، باید nano را با استفاده از دسترسی‌های super user اجرا کنید یا مجوزهای فایل را تغییر دهید، همان‌طور که در فصل 14، "Setting Ownership and Permissions" توضیح داده شده است.

نکته: اگر نام فایل موجود را اشتباه تایپ کنید، nano یک فایل خالی را نشان خواهد داد. بنابراین، اگر به جای فایلی که انتظار داشتید، یک فایل خالی مشاهده کردید، ممکن است نام فایل را اشتباه تایپ کرده باشید.

## Using Text Editor Conventions

هر ادیتور متنی دارای قواعد خاص خود برای نمایش اطلاعات بر روی صفحه، دستکاری متن و غیره است. بیشتر ادیتورهای متنی تکست مود تا حدی مشابه هستند - به عنوان مثال، یک یا چند خط در بالای یا پایین

به طور معمول اطلاعات خلاصه یا دستورات مختصر را نشان می‌دهند. شکل 10.1 و شکل 10.2 این اطلاعات را برای nano نشان می‌دهند که شامل موارد زیر است:

**Title Bar**: اولین خط، نوار عنوان است. این خط شامل شماره نسخه nano، نام فایل در حال ویرایش و وضعیت تغییرات می‌باشد

**Status Bar**: سومین خط از پایین برای اطلاعات وضعیت و تعاملات با کاربر اختصاص یافته است. این خط شما را برای اطلاعاتی مانند نام فایل برای ذخیره هنگام ذخیره‌سازی سند یا عبارات مورد جستجو در سند هنگام انجام عملیات جستجو راهنمایی می‌کند.

**Shortcut List**: دو خط پایین خلاصه‌ای از برخی از رایج‌ترین عملیات‌ها به همراه کلیدهای ترکیبی که آن‌ها را فعال می‌کنند، نمایش می‌دهد.

نکته: در مستندات nano، یک علامت (^) که قبل از یک حرف می‌آید به یک کاراکتر کنترل اشاره دارد. در این کتاب، چنانی ترکیب کلیدهایی با Ctrl+ به جای ^ نشان داده می‌شوند.

علاوه بر کاراکترهای کنترلی، nano از متاکاراکترها برای فعال‌سازی برخی از عملکردها استفاده می‌کند. این ترکیب کلیدها از کلیدهای Alt، Esc یا Meta (بسته به پیکربندی کیبورد شما) به همراه یک کلید دیگر استفاده می‌کنند. در مستندات nano، توالی‌های متا با M-k نشان داده می‌شوند، جایی که k یک کلید است. به عنوان مثال، ?-M توالی کلیدی برای حرکت به خط آخر سند است. توجه داشته باشید که این توالی‌ها متفاوت از توالی‌های کلید Ctrl هستند؛ به این معنا که شما باید Esc (یا Alt یا Meta) را فشار داده، رها کنید و سپس کلید علامت سوال (?) را به همراه شیفت فشار دهید تا به خط آخر سند بروید.

نکته: معمولاً، کیبوردهای مدرن کلید Meta ندارند (داسکی‌منت‌های سازنده کیبورد خود را چک کنید). به عنوان جایگزین، ابتدا سعی کنید از کلید Esc به جای کلید Meta برای توالی‌های متا در nano استفاده کنید. اگر این کار نکرد، از کلید Alt به عنوان کلید Meta استفاده کنید.

## Exploring Basic nano Text-Editing Procedures

برای یادگیری nano، به ایجاد و ویرایش فایل pets.txt توجه کنید، همانطور که در شکل 10.1 نشان داده شده است. می‌توانید به راحتی این مثال را با سیستم خود اجرا کنید. ابتدا فایل را با نوع حیوانات خانگی پر کنید، همانطور که در فهرست 10.1 نشان داده شده است. پس از اینکه داده‌های فایل را روی دیسک ذخیره کردید، با استفاده از ویرایشگر متن، یک نوع حیوان خانگی جدید اضافه کنید.

Listing 10.1: Sample pets.txt file

dog

cat

bird

fish

برای این مثال، اولین قدم برای استفاده از nano این است که آن را اجرا کنید و یک فایل خالی به نام pets.txt بسازید. این نشان می‌دهد که شما یک فایل جدید و خالی به نام pets.txt ایجاد کرده‌اید:

1. یک برنامه ترمینال باز کنید و در دایرکتوری خانه خود، عبارت nano pets.txt را تایپ کرده و کلید Enter را فشار دهید. شما باید کلمات title bar و همچنین کلمات New File را در مرکز status bar مشاهده کنید. این نشان می‌دهد که شما یک فایل جدید و خالی به نام pets.txt ایجاد کرده‌اید.

2. کلمه dog را تایپ کرده و برای افزودن نوع حیوان خانگی اول به فایل، کلید Enter را فشار دهید. ادامه دهید با نوع حیوانات خانگی از فهرست 10.1 تا زمانی که همه چهار نوع حیوان خانگی را اضافه کرده باشید.

در این مرحله، نتیجه شما باید شبیه به شکل 10.1 (که قبلًا نشان داده شده) باشد که nano در حال ویرایش فایل pets.txt است. شما می‌توانید یک ورودی جدید به pets.txt اضافه کنید. اولین راه این است که یک خط خالی جدید ایجاد کنید و ورودی را به صورت دستی تایپ کنید. می‌توانید این کار را به صورت زیر انجام دهید:

1. کلیدهای جهتنما را فشار دهید تا مکان‌نما را بر روی حرف f در کلمه fish منتقل کنید.

2. کلید Enter را فشار دهید. این عمل یک خط جدید بین خطوط bird و fish باز می‌کند.

3. یک بار کلید جهتنمای بالا را فشار دهید تا مکان‌نما را به خط خالی جدید منتقل کنید.

4. کلمه reptile را تایپ کنید و کلید Enter را فشار ندهید.

راه دوم برای ایجاد یک ورودی جدید نشان می‌دهد که چگونه می‌توانید متن را در ویرایشگر کپی، کات و پیست کنید:

1. با استفاده از کلیدهای جهت‌دار، نشانگر را به ابتدای خط reptile که تازه ایجاد کرده‌اید ببرید؛ باید نشانگر روی حرف r در کلمه reptile باشد.

2. کلیدهای M-6 را فشار دهید. (دومین کلید عدد 6 است، نه حرف r، و کلید Meta ممکن است Alt یا باشد، بسته به پیکربندی صفحه کلید شما.) شاید به نظر نرسد که چیزی اتفاق افتاده است، اما این کلید، خطی که نشانگر روی آن قرار دارد را به یک بافر کپی می‌کند. اکنون باید نشانگر شما روی حرف f در خط fish باشد.

3. کلیدهای **Ctrl+U** را فشار دهید. این کلید محتویات بافر را در مکان فعلی در فایل جایگذاری می‌کند. باید ببینید که نوع حیوان خانگی **reptile** در دو خط نمایش داده می‌شود.

4. از کلیدهای جهت‌دار برای حرکت دادن نشانگر به ابتدای خط دوم **reptile** که تازه ایجاد کرده‌اید استفاده کنید؛ باید نشانگر روی حرف **r** در کلمه **reptile** باشد.

5. کلیدهای **Ctrl+K** را فشار دهید تا کل خط دوم **reptile** برش داده شود. نشانگر باید در ابتدای خط **fish** باشد.

6. کلیدهای **Ctrl+N** را فشار دهید تا به خط بعدی در فایل بروید. نشانگر شما باید زیر خط **fish** باشد.

7. برای افزودن نوع حیوان خانگی دیگر، کلمه **rodent** را تایپ کنید.

نکته: برای ذخیره فایل جدید **pets.txt** که با استفاده از نانو ایجاد کرده‌اید، بخش بعدی به نام "Changes from nano" را بخوانید.

می‌توانید تغییرات اضافی را به همین روش انجام دهید. اگرچه نانو نسخه **GUI** ندارد، اصول اصلی آن مانند ادیتورهای **GUI** هستند؛ شما فقط باید کلیدهای میانبر را برای فعال کردن ویژگی مورد نظرتان بدانید.

вшردن کلیدهای **Ctrl+G** داکیومنت های راهنمای نانو را نمایش می‌دهد که خلاصه ویژگی‌های برنامه را شامل می‌شود. این مورد می‌تواند مفید باشد زمانی که شما در حال آشنایی با ویرایشگر هستید. برخی از ویژگی‌های اضافی که ممکن است بخواهید استفاده کنید عبارتند از:

**PageUp** **PageDown**: شما می‌توانید از کلیدهای جهت‌نما، **Home**، و **End** برای حرکت نشانگر استفاده کنید که در سایر ادیتورها نیز متداول هستند. برای حرکت به ابتدای فایل، کلیدهای **6-M** را فشار دهید و برای حرکت به انتهای فایل، کلیدهای **-M** را فشار دهید. (به یاد داشته باشید که **M** به معنای کلید **Meta** است که در بخش «Using Text Editor Conventions» این فصل پوشش داده شده است).

**Copy or Move Multiple Lines**: اگر نیاز دارید چندین خط متوالی را کپی یا جابجا کنید، می‌توانید عملیات **Ctrl+K** یا **Ctrl+M** را تکرار کنید؛ **nano** تمامی خطوط را که کپی یا کات کرده اید را نگه می‌دارد تا وقتی **U** را فشار دهید، همه آنها دوباره پیست شوند.

**Insert a File**: فشردن **Ctrl+R** یا **5F** به شما امکان می‌دهد تا یک فایل دیگر را در موقعیت فعلی نشانگر به فایل جاری وارد کنید.

**Search for a String**: فشردن **Ctrl+W** یا **6F** ویژگی جستجو را فعال می‌کند. زمانی که این ویژگی فعال شود، **nano** از شما می‌خواهد یک عبارت جستجو وارد کنید. آن را تایپ کرده و کلید **Enter** را فشار دهید، سپس اولین مورد از آن عبارت جستجو را در فایل پیدا می‌کند. وقتی دوباره **Ctrl+W** یا **6F** را فشار دهید، عبارت جستجوی پیش‌فرض آخرین عبارت استفاده شده خواهد بود، بنابراین می‌توانید با فشردن **Ctrl+W** یا

F6 و سپس Enter برای هر عملیات جستجو، چندین بار همان عبارت جستجو کنید. به طور جایگزین، M-W آخرین جستجو را تکرار می‌کند.

**Replace a String**: می‌توانید یک رشته را با رشته دیگری جایگزین کنید با فشردن \ Ctrl+R یا M-R. برنامه از شما می‌خواهد یک عبارت جستجو و عبارتی که باید جایگزین آن شود را وارد کنید. سپس جستجو آغاز می‌شود و nano از شما می‌خواهد تا هر جایگزینی را تأیید کنید. اگر می‌خواهید تمام موارد بدون پرسش جایگزین شوند، می‌توانید در اولین پرسش کلید A را فشار دهید.

## Saving Your Changes from nano

بعد از اینکه تغییراتی در یک فایل متنی ایجاد کردید، احتمالاً می‌خواهید آنها را ذخیره کنید. یکی از راههای انجام این کار استفاده از گزینه Ctrl+O است. (حرف O، نه عدد ۰). وقتی این کلید را فشار می‌دهید، nano از شما می‌پرسد:

Write Selection to File:

به طور معمول، پaramپت شامل نام اصلی فایل خواهد بود، بنابراین می‌توانید کلید Enter را فشار دهید تا فایل با استفاده از همان نام ذخیره شود. اگر می‌خواهید از نام دیگری استفاده کنید، می‌توانید نام قدیمی را حذف کرده و نام جدیدی را تایپ کنید. اگر nano را بدون مشخص کردن نام فایل راه اندازی کرده‌اید، می‌توانید در این درخواست نامی تایپ کنید.

راه دیگر برای ذخیره فایل، فشار دادن Ctrl+X است. این فرمان از nano خارج می‌شود، اما اگر فایل را تغییر داده‌اید، پaramپت زیر را نشان می‌دهد:

Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?

در این پaramپت، حرف y را تایپ کنید تا فایل ذخیره شود. برنامه سپس پaramپت نام فایل را که اگر O را فشار داده بودید، نمایش می‌دهد، بنابراین می‌توانید نام فایل را در صورت تمایل تغییر دهید. پس از اینکه nano فایل را ذخیره کرد، خاتمه می‌یابد.

اگر در سیستم خود مراحل ایجاد و ویرایش فایل pets.txt را که قبلًا در این فصل توضیح داده شده، دنبال می‌کردید، این مراحل را برای ذخیره تغییرات و خروج از ویرایشگر متن nano انجام دهید:

1. در nano که فایل pets.txt نمایش داده می‌شود، کلید O Ctrl+O را فشار دهید تا فرآیند ذخیره فایل آغاز شود. در status bar باید عبارت "File Name to Write: pets.txt" را نشان دهد.

2. کلید Enter را فشار دهید تا فایل ذخیره شود. باید چیزی مشابه عبارت "Wrote 6 lines" را در status bar ببینید.

3. کلید X را فشار دهید تا از ویرایشگر متن nano خارج شده و به کامند لاین بازگردید.

برای کسب تجربه بیشتر، تلاش کنید که فایل pets.txt را دوباره با استفاده از nano ویرایش کرده و تغییرات مختلفی اعمال کنید و دستورات ادیت مختلف را امتحان کنید. حتماً از فیچر راهنمای نیز استفاده کنید، با فشردن Ctrl+G به آن دسترسی پیدا کنید و با فشردن Ctrl+X از مستندات راهنمای خارج شوید.

## Editing Files with vi

vi اولین ویرایشگر متنی با نمایش تمام صفحه برای سیستم عامل یونیکس بود. vi طوری طراحی شده که به اندازه کافی کوچک باشد تا بتواند در سیستم‌های قدیمی با بوت اضطراری بر اساس دیسک‌های فلاپی جای بگیرد. بعدها، نسخه‌ای جدید با تعدادی اصلاحات به نام "vi improved" یا vim ایجاد شد. با وجود اینکه بیشتر توزیع‌های لینوکس با vim ارائه می‌شوند، همچنان به عنوان ویرایشگر vi شناخته می‌شود. vim با ویرایشگر vi سازگاری به روزرسانی دارد و دستور برای اجرای vim معمولاً vi است، اگرچه برخی از توزیع‌ها دستور vim نیز دارند. اطلاعات ارائه شده در این فصل برای هر دو vi و vim قابل اعمال است.

با اینکه با vi می‌توان به خوبی فایل‌های کانفیگ را ویرایش کرد، اما ویژگی برجسته‌اش در ویرایش فایل‌های برنامه‌ای مانند اسکریپت‌های شل است. بنابراین، یادگیری ویرایشگر vi به شما کمک زیادی خواهد کرد، اگرچه بسیاری آن را پیچیده‌ترین تکست ادیتور برای استفاده می‌دانند.

## Understanding vi Modes

برای استفاده از vi، ابتدا باید با حالت‌هایی که در آن عمل می‌کند آشنا شوید. سپس می‌توانید شروع به یادگیری روش‌های ویرایش متنی که vi ارائه می‌دهد، نمایید. در هر لحظه، vi در یکی از سه حالت زیر اجرا می‌شود:

**Command Mode:** این حالت دستوراتی را قبول می‌کند که معمولاً به صورت حروف تکی وارد می‌شوند. به عنوان مثال، دستور a و A هر دو وارد حالت درج می‌شوند، با این تفاوت که به شکل‌های مختلفی اعمال می‌شوند که به زودی شرح داده می‌شود، و دستور 0 یک خط را در زیر خط فعلی باز می‌کند.

**Ex Mode:** برای مدیریت فایل‌ها (شامل ذخیره کردن فایل فعلی و اجرای برنامه‌های خارجی)، شما از حالت ex استفاده می‌کنید. برای ورود به حالت ex از حالت دستورات، کافی است که دو نقطه (: ) را تایپ کنید، که معمولاً به آن دستور ex که می‌خواهید استفاده کنید، اضافه می‌شود. پس از اجرای دستور حالت ex، vi به طور خودکار به حالت دستورات باز می‌گردد.

**Insert Mode:** در این حالت، شما متن را وارد می‌کنید. اکثر دکمه‌های فشرده شده باعث نمایش متن در صفحه می‌شوند. یک استثناء مهم کلید Esc است که حالت وارد کردن متن را خاتمه داده و به mode باز می‌گردد.

متأسفانه، اصطلاحات مربوط به حالات vi به بهترین شکل ممکن یکپارچه نیستند. به عنوان مثال، command mode گاهی اوقات به عنوان normal mode شناخته می‌شود و Insert Mode گاهی اوقات به عنوان colon commands entry mode یا ex-mode edit mode نامیده می‌شود. حالت ex-mode اغلب به عنوان توصیف نمی‌شود و به عنوان یک mode معرفی نمی‌شود.

## Checking your vi/vim Editor Package

سیستم لینوکس شما ممکن است به طور پیشفرض بسته کامل vim را نداشته باشد. به عنوان مثال، توزیع شما ممکن است فقط با بسته vim.tiny یا vim.minimal نصب شده باشد. با این بسته‌ها، هنوز می‌توانید به یک نسخه از vi دسترسی پیدا کنید، اما ممکن است به برخی از ویژگی‌های مختلف این ادیتور دسترسی کامل نداشته باشید، از جمله برخی از آنچه که در این فصل توضیح داده شده است. اگر قصد دارید vi را یاد بگیرید و به درستی استفاده کنید، باید بسته کامل vim را نصب کنید.

برای بررسی سیستم خود، دستور `type vi` را در کامند لاین وارد کنید. باید نام برنامه (vi یا vim) به همراه مکان دایرکتوری آن را مشاهده کنید، مشابه این نمونه:

```
$ type vi  
vi is hashed (/usr/bin/vi)
```

زمانی که نام برنامه و مکان آن را دارید، تایپ کنید `readlink -f /location/program` تا بررسی کنید که آیا برنامه به یک برنامه دیگر متصل است. (در فصل 7، "Managing Files") شما ممکن است چیزی شبیه به زیر دریافت کنید:

```
$ readlink -f /usr/bin/vi  
/usr/bin/vim.tiny
```

در ادامه، شما همچنان نیاز به بررسی بسته نرم‌افزاری دارید که برنامه را فراهم کرده است. (در برخی موارد، نام برنامه نشان نمی‌دهد کدام بسته نرم‌افزاری آن را فراهم کرده است). از ابزار مناسب بسته نرم‌افزاری برای تعیین این اطلاعات لازم استفاده کنید. مثال زیر بر روی توزیع Linux Mint است، بنابراین دستور `dpkg -S` به همراه دسترسی به دستورهای کاربر root (دریافت دسترسی به کاربر روت در فصل 12 به طور دقیق تر شرح داده شده است) استفاده می‌شود.

```
$ sudo dpkg -S /usr/bin/vim.tiny  
[sudo] password for christine:  
vim-tiny: /usr/bin/vim.tiny
```

در این مورد مشاهده می‌شود که بسته vim-tiny نصب شده است. اگر بینید که بسته vim vim-enhanced یا vim-basic، vim-runtime نصب شده است، نباید مشکلی در استفاده از دستورات ویرایشگر vi در این فصل داشته باشید. با این حال، اگر هیچ‌کدام از این بسته‌ها را پیدا نکردید، باید بسته vim را نصب کنید تا بتوانید ادامه این فصل را دنبال کنید. برای مثال قبلی روی توزیع Linux Mint، شما می‌توانید دستور `sudo apt-get install vim` را وارد کنید تا بسته کامل vim را نصب کنید. بر روی توزیع Fedora Workstation دستور `sudo yum install vim` را وارد کنید. (نصب بسته‌ها در فصل 9 شرح داده شده است).

اکنون می‌توانید شروع به یادگیری روش‌های ادیت تکست که ادیتور vi پیاده‌سازی می‌کند کنید. همچنین نحوه ذخیره فایل‌ها و خروج از vi را بررسی خواهید کرد.

## Exploring Basic vi Text-Editing Procedures

به عنوان یک روش برای یادگیری vi، در نظر بگیرید که به وظیفه‌ای مانند ایجاد و ویرایش فایل pets.txt مشغول شوید، که در بخش قبلی با عنوان "Exploring Basic nano Text-Editing Procedures". توضیح داده شده است. لیست 10.1 (در آن بخش قبلی) فایل pets.txt را نشان می‌دهد که برای مثال بعدی باید ایجاد شود.

نکته: اگر پیش‌تر فایل pets.txt را ایجاد کرده‌اید و می‌خواهید در این بخش پیش بروید، ابتدا فایل را با تایپ دستور `rm pets.txt` و فشردن دکمه Enter حذف کنید.

برای این مثال، اولین گام برای استفاده از ویرایشگر vi این است که آن را راه‌اندازی کنید و فایل خالی به نام pets.txt ایجاد کنید، به این صورت:

1. یک برنامه ترمینال باز کنید و در دایرکتوری home، دستور vi pets.txt را وارد کنید و Enter را بزنید. شما باید کلمات [New File] "pets.txt" را در پایین مشاهده کنید. این نشان می‌دهد که شما یک فایل خالی جدید به نام pets.txt ایجاد کرده‌اید.

2. در حالت دستور، کلید i را بزنید تا وارد حالت insert mode شوید. این حالت باید با کلمه --INSERT-- در پایین نشان داده شود.

3. کلمه dog را تایپ کنید و Enter را بزنید تا نوع اول حیوان خانگی را به فایل اضافه کنید. ادامه دادن با انواع حیوان خانگی از لیست 10.1 تا زمانی که همه چهار نوع حیوان را اضافه کرده‌اید.

4. کلید Esc را بزنید تا از حالت وارد خارج شوید. توجه داشته باشید که پیام --INSERT-- دیگر نمایش داده نمی‌شود. ویرایشگر vi در حالت command mode است.

5. را تایپ کنید تا وارد حالت ex شوید، و دستور را با تایپ wq تکمیل کرده و Enter را بزنید. این دستور محتویات بافر vi را به فایل pets.txt می‌نویسد (w)، ویرایشگر vi را خروجی می‌دهد (q) و شما را به کامند لاین باز می‌گرداند.

6. دوباره فایل pets.txt را در ویرایشگر vi با تایپ Enter و vi pets.txt باز کنید.

می‌بایست نتیجه شما مشابه شکل 10.3 باشد که vi فایل pets.txt را در command mode نشان می‌دهد. همانطور که در شکل 10.3 نشان داده شده است، برخی سیستم‌ها یک خط از موجودی (~) را در طرف چپ صفحه نمایش می‌دهند تا انتهای فایل را نشان دهند. فایل در شکل 10.3 به تازگی بارگذاری شده است و به همین دلیل خط پایین نمایش آخرین دستور را نشان می‌دهد - یک دستور بارگذاری ضمنی که 4 خط و 18 کاراکتر از فایل pets.txt را بارگذاری کرده است.

```
christine@Kaylee-FF ~
File Edit View Search Terminal Help
dog
cat
bird
fish
~
"pets.txt" 4 lines, 18 characters
```

Figure 10.3 The last line of a vi display is a status line that shows messages from the program.

نکته: ممکن است جذاب باشد که از شورتکات های نرم افزارهای ادیتور در ادیتور تکست مود استفاده کنید، مانند استفاده از Ctrl+S برای ذخیره فایل. این عملکرد مناسبی نیست، زیرا این ترکیب کلید ممکن است باعث قفل شدن ترمینال شما شود. اگر این کار را کرده‌اید، می‌توانید ترمینال خود را با فشردن Ctrl+Q بازگشایی کنید.

همانطور که در نانو نیز می‌توانید با استفاده از وی، یک ورودی جدید به pets.txt اضافه کنید، این کار را می‌توان با تایپ کردن یک خط جدید به صورت کامل یا با تکرار یک خط موجود و اصلاح آن انجام داد. برای تایپ یک خط جدید، این مراحل را دنبال کنید:

1. مکان نما را با استفاده از کلیدهای فلش به ابتدای خط bird منتقل کنید.
2. کلید O (حرف O، نه عدد 0) را فشار دهید. این کار یک خط جدید را به طور فوری زیر خط فعلی باز می‌کند، مکان نما را به آن خط منتقل کرده و وارد حالت ادیت می‌شود.
3. متن جدید را تایپ کنید: reptile
4. برای بازگشت از حالت ادیت به حالت command mode، کلید Esc را فشار دهید.

به منظور تمرین در انجام تغییرات با ویرایش یک ورودی موجود، این مراحل را دنبال کنید:

1. با استفاده از کلیدهای فلش، مکان نمایشگر را به ابتدای خط `reptile` که به تازگی ایجاد کرده‌اید، ببرید. شما باید نمایشگر را بر روی حرف `r` از `reptile` ببینید.

2. شما در حال حاضر یک خط متنی را کپی (yank) می‌کنید. اصطلاح کپی در بسیاری از ویرایشگرهای تکست مود استفاده می‌شود - شما متن را به یک بافر کپی می‌کنید که بعداً می‌توانید آن را به فایل برگردانید. برای کپی متن، از دستور `yy` به همراه تعداد خطوط مورد نظر برای کپی استفاده کنید. بنابراین، تایپ کنید `yy` (هنوز `Enter` را فشار ندهید). به نظر نمی‌رسد که چیزی انجام شده باشد، اما این دستور کل خط را که نمایشگر روی آن قرار دارد، به بافر کپی می‌کند.

3. با استفاده از کلیدهای جهت‌دهی، نمایشگر را به خط `bird` ببرید که خط قبل از محلی است که می‌خواهید خط جدید ظاهر شود.

4. کلید `p` را تایپ کنید (دوباره `Enter` را فشار ندهید). `vi` محتوای بافر (`reptile`) را شروع کرده از خط بعد از نمایشگر می‌کند. حالا فایل باید دو خط یکسان `reptile` داشته باشد. نمایشگر باید در ابتدای اولین خط باشد.

5. نمایشگر را به حرف `z` در کلمه `reptile` در خطی که به تازگی `paste` کرده‌اید، اگر هنوز اینجا نیست، ببرید. شما در حال حاضر در حال حذف این خط هستید.

6. دستور `dd` به همان شکلی که `yy` عمل می‌کند کار می‌کند، با این تفاوت که خطوط را هم حذف می‌کند و به بافر کپی می‌کند. برای حذف خط `reptile`، `dd` را تایپ کنید. حالا فایل باید فقط یک خط `reptile` داشته باشد.

7. فایل را ذخیره و خروج کنید با تایپ `ZZ`. این دستور معادل `wq` است.

نکته: اگر نیاز به تغییر کد گذاری `end-of-line` یک فایل تکست از روش ویندوز به روش یونیکس/لینوکس دارید، فایل را در ویرایشگر `vi` باز کنید و این دستور را تایپ کنید: `set ff=unix`:

بسیاری از دستورات اضافی موجود هستند که ممکن است در برخی موارد مفید باشند. در ادامه، تعدادی از این دستورات را معرفی می‌کنیم:

**Change Case**: فرض کنید که نیاز دارید که حروف یک کلمه را در یک فایل به صورت حروف بزرگ یا کوچک تغییر دهید. به جای وارد کردن حالت ویرایش و تایپ مجدد کلمه، می‌توانید از کلید `Tilde (~)` در حالت `command mode` استفاده کنید تا حالت حروف را تغییر دهید. نشانگر را بر روی اولین حرفی که می‌خواهید تغییر دهید قرار دهید، و تکراراً کلید `~` را فشار دهید تا کار انجام شود.

**Undo**: برای لغو هر تغییر، در حالت `command mode`، حرف `u` را تایپ کنید.

**Open Text**: در حالت `command mode`، تایپ کردن حرف `O` (بزرگ یا کوچک) متن را باز می‌کند؛ به این معنا که یک خط جدید را به طور فوری زیر خط فعلی اضافه کرده و وارد حالت `entry mode` در آن خط می‌شود.

**Search**: برای جستجوی متن به جلو در یک فایل، در حالت فرمان، ابتدا علامت / را تایپ کنید، سپس فوراً متن مورد نظر را بنویسید که می‌خواهید جستجو کنید. اگر می‌خواهید به جای عقب جستجو کنید، می‌توانید از علامت ? استفاده کنید.

**Change Text**: دستور c متن را از حالت فرمان تغییر می‌دهد. شما می‌توانید آن را مانند دستورهای d یا y فراخوانی کنید، به عنوان مثال cw برای تغییر کلمه بعدی یا cc برای تغییر کل خط.

**Go to a Line**: کلید G شما را به خطی که شما مشخص می‌کنید می‌برد. کلید H مکان کرسر را به بالاترین خط صفحه می‌برد. کلید L کرسر را به پایین‌ترین خط صفحه می‌برد.

**Replace Globally**: برای جایگزینی تمام وقوع‌های یک رشته با رشته‌ای دیگر، کافی است دستور زیر را در حالت ex-mode وارد کنید: `original/g`، جایی که `original` رشته اصلی است و `replacement` رشته جایگزین آن است. اگر می‌خواهید این تغییر را بر روی یک محدوده خطوط انجام دهید، می‌توانید `%` را با شماره خط شروع و پایان جایگزین کنید، که با کاما از هم جدا می‌شوند.

نکته: g/ در انتهای دستور لازم نیست اگر رشته original فقط یک بار در هر خط فایل وجود داشته باشد.

vi امکانات بسیار بیشتری از آنچه در اینجا ارائه شده دارد؛ این ویرایشگر بسیار قدرتمند است و برعی از کاربران لینوکس به شدت به آن وابسته هستند. کتابهای کاملی درباره vi نوشته شده‌اند. برای اطلاعات بیشتر می‌توانید به یکی از این کتاب‌ها یا به صفحه وب [www.vim.org](http://www.vim.org) مراجعه کنید.

## Saving Your Changes from vi

برای ذخیره تغییرات در فایل بدون خروج از ویرایشگر، در حالت فرمان w: را تایپ کنید. این کار وارد حالت ex شده و فرمان ex-mode به نام w را اجرا می‌کند که فایل را با نام فایلی که هنگام راهاندازی vi مشخص کرده‌اید ذخیره می‌کند. فرمان‌های مرتبط عملکردهای دیگری را امکان‌پذیر می‌سازند:

**ویرایش یک فایل جدید**: فرمان e: یک فایل جدید را برای ویرایش باز می‌کند. به عنوان مثال،

دستور /etc/inittab e: فایل /etc/inittab را برای ویرایش بارگذاری می‌کند. vi فایل جدیدی را بارگذاری نمی‌کند مگر اینکه فایل فعلی از آخرین تغییراتش ذخیره شده باشد یا اینکه شما بعد از e: علامت تعجب (!) بگذارید. به خاطر داشته باشید که اگر از علامت تعجب استفاده کنید، هرگونه تغییری که در فایل اصلی داده‌اید از بین خواهد رفت.

**اضافه کردن یک فایل موجود**: فرمان a: محتويات یک فایل قدیمی را در یک فایل موجود وارد می‌کند و آن را به فایل موجود اضافه می‌کند.

**اجرای یک فرمان خارجی**: فرمان حالت ex با !: فرمان خارجی را که مشخص می‌کنید اجرا می‌کند. به عنوان مثال، تایپ کردن ls !: فرمان ls را اجرا می‌کند و به شما امکان می‌دهد تا ببینید چه فایل‌هایی در دایرکتوری فعلی موجود هستند.

خروج: از دستور `q`: برای خروج استفاده کنید. همانطور که با دستور `e`:، این دستور تا زمانی که تغییرات ذخیره نشده باشند کار نخواهد کرد مگر اینکه با علامت تعجب (`مانند !q!`):) به خاطر داشته باشید که اگر از علامت تعجب استفاده کنید، هرگونه تغییرات انجام شده در فایل اصلی از بین خواهد رفت.

شما می‌توانید دستورات `ex` را مانند این‌ها ترکیب کنید تا چندین عمل را به ترتیب انجام دهید. به عنوان مثال، همانطور که قبلًا نشان داده شد، تایپ کردن `Wq`: تغییرات را ذخیره کرده و سپس از `vi` خارج می‌شود.

# CHAPTER 11

## Creating Scripts

اسکریپت یک برنامه نوشته شده در یک زبان تفسیری است، که معمولاً با یک شل یا برنامه دیگر مرتبط است که هدف اصلی آن چیزی غیر از زبان تفسیری است. در لینوکس، بسیاری از اسکریپتها، اسکریپتهاشی شل هستند که با Bash یا شل دیگری مرتبط هستند. (اگر با فایلهای batch در DOS یا ویندوز آشنا هستید، اسکریپتها هدف مشابهی دارند.) شما می‌توانید اسکریپتهاشی شل بنویسید تا وظایف تکراری و خسته‌کننده را خودکار کنید یا وظایف جدید و پیچیده را انجام دهید. بسیاری از عملکردهای راه‌اندازی لینوکس توسط اسکریپتها انجام می‌شوند، بنابراین تسلط بر اسکریپتنویسی به شما کمک می‌کند تا فرآیند راه‌اندازی را مدیریت کنید.

این فصل به بررسی اسکریپتهاشی شل Bash می‌پردازد و با ایجاد یک فایل اسکریپت جدید آغاز می‌شود. سپس چندین ویژگی مهم اسکریپتنویسی که به شما کمک می‌کند تا وظایف اسکریپتنویسی پیچیده‌تر را انجام دهید، توصیف می‌کند.

نکته: مانند هر کار برنامه‌نویسی، اسکریپتنویسی شل می‌تواند بسیار پیچیده باشد. در نتیجه، این فصل فقط کمی از آنچه که می‌توانید از طریق اسکریپتنویسی شل انجام دهید را پوشش می‌دهد. برای اطلاعات بیشتر به کتابی در این زمینه، مانند کتاب زیر مراجعه کنید.

"Linux Command Line and Shell Scripting Bible, Third Edition" (Wiley, 2015)

### Beginning a Shell Script

اسکریپتهاشی شل فایلهای متند، بنابراین آنها را در ویرایشگرهای متند مانند vi، nano یا pico که در فصل 10، "Editing Files" توضیح داده شده است، ایجاد می‌کنید. نباید از برنامه‌هایی مانند LibreOffice Writer برای ایجاد اسکریپتهاشی شل استفاده کنید، زیرا به طور پیش‌فرض آنها را به عنوان کدهای باینری به سند نهایی اضافه می‌کنند تا فونت‌ها، اندازه‌های فونت و کاراکترهای خاص را نشان دهند. این کدهای باینری شل لینوکس را گیج می‌کنند.

یک اسکریپت شل با خطی شروع می‌شود که شل مورد استفاده برای اجرای آن را مشخص می‌کند، مانند نمونه زیر:

```
#!/bin/bash
```

دو کاراکتر اول یک کد ویژه هستند که به کرنل لینوکس می‌گویند این یک اسکریپت است و باید از بقیه خط به عنوان مسیر برنامه‌ای که اسکریپت را تفسیر می‌کند استفاده کند. (این خط گاهی shebang یا hashbang، hashpling، pound bang، یا pound bang نامیده می‌شود.) زبان‌های اسکریپتنویسی شل از علامت هشتگ (#) به عنوان کاراکتر توضیح استفاده می‌کنند، بنابراین ابزار اسکریپت این خط را نادیده می‌گیرد، هرچند که کرنل آن را نادیده نمی‌گیرد. در اکثر سیستم‌ها، /bin/sh یک لینک سمبولیک است که به /bin/bash اشاره

می‌کند، اما می‌تواند به شل دیگری نیز اشاره کند. مشخص کردن اسکریپت به عنوان استفاده از /bin/sh تضمین می‌کند که هر سیستم لینوکسی یک برنامه شل برای اجرای اسکریپت خواهد داشت، اما اگر اسکریپت از هر ویژگی خاصی از یک شل خاص استفاده کند، باید آن شل خاص را مشخص کنید—برای مثال، اسکریپت از /bin/tcsh /bin/bash یا /bin/sh استفاده کنید.

نکته: این فصل اسکریپتهاشی شل Bash را توضیح می‌دهد. اسکریپتهاشی ساده Bash می‌توانند در شلهای دیگر نیز اجرا شوند، اما اسکریپتهاشی پیچیده‌تر بیشتر احتمال دارد که مربوط به شل خاص باشند.

زمانی که نوشتمن اسکریپت شل را به پایان رساندید، باید آن را طوری تغییر دهید که قابل اجرا باشد. این کار را با فرمان `chmod` انجام می‌دهید که در فصل 14، "Setting Ownership and Permissions"، به تفصیل توضیح داده شده است. فعلًاً بدانید که از گزینه `a+x` برای افزودن مجوز اجرا برای همه کاربران استفاده می‌کنید. برای مثال، برای قابل اجرا کردن فایلی به نام `my-script`، فرمان زیر را صادر می‌کنید:

```
$ chmod a+x my-script
```

سپس می‌توانید اسکریپت را با تایپ نام آن اجرا کنید، ممکن است لازم باشد /. را پیش از نام اسکریپت بیاورید تا لینوکس اسکریپت را در دایرکتوری جاری اجرا کند و نه اینکه مسیر جاری را جستجو کند. اگر اسکریپت را قابل اجرا نکرده‌اید، هنوز می‌توانید با اجرای برنامه شل و پس از آن نام اسکریپت (مانند bash) اسکریپت را اجرا کنید، اما به طور کلی بهتر است اسکریپت را قابل اجرا کنید. اگر اسکریپت یکی از اسکریپتهاشی است که به طور مرتباً اجرا می‌کنید، ممکن است بخواهید آن را به مکانی در مسیر خود، مانند /usr/local/bin، منتقل کنید. هنگامی که این کار را انجام می‌دهید، نیازی به تایپ مسیر کامل یا انتقال به دایرکتوری اسکریپت برای اجرای آن نخواهید داشت؛ فقط می‌توانید my-script را تایپ کنید.

## Using Commands

یکی از ویژگی‌های اساسی اسکریپتهاشی شل، توانایی اجرای دستورات است. شما می‌توانید از دستورات داخلی شل و دستورات خارجی استفاده کنید؛ یعنی می‌توانید برنامه‌های دیگر را به عنوان دستورات اجرا کنید. بیشتر دستورات که در یک شل تایپ می‌کنید، دستورات خارجی هستند؛ این‌ها برنامه‌هایی هستند که در دایرکتوری‌های /usr/bin و دیگر دایرکتوری‌های مسیر شما قرار دارند. شما می‌توانید این برنامه‌ها و همچنین دستورات داخلی را با درج نام‌های آن‌ها در اسکریپت اجرا کنید. همچنین می‌توانید پارامترهایی برای این برنامه‌ها در اسکریپت مشخص کنید. برای مثال، فرض کنید می‌خواهید یک اسکریپت داشته باشید که دو پنجره xterm و برنامه ایمیل KMail را راهاندازی کند. فهرست 11.1 یک اسکریپت شل را نشان می‌دهد که این هدف را محقق می‌کند.

Listing 11.1: A simple script that launches three programs

```
#!/bin/bash  
  
/usr/bin/xterm &  
  
/usr/bin/xterm &  
  
/usr/bin/kmail &
```

به جز خط اول که آن را به عنوان یک اسکریپت شناسایی می‌کند، این اسکریپت شبیه به دستورات معمولی است که برای دستیابی به این کارها به صورت دستی تایپ می‌کنید، به جز یک نکته: اسکریپت مسیرهای کامل هر برنامه را فهرست می‌کند. این معمولاً کاملاً ضروری نیست، اما فهرست کردن مسیر کامل اطمینان می‌دهد که اسکریپت برنامه‌ها را حتی اگر متغیر محیط PATH تغییر کند، پیدا می‌کند. از سوی دیگر، اگر فایل‌های برنامه جابجا شوند (مثلاً به دلیل بهروزرسانی بسته‌ای که از آن‌ها نصب شده‌اند و بسته‌بندی‌کننده تصمیم بگیرد آن‌ها را جابجا کند)، اسکریپت‌هایی که از مسیرهای کامل استفاده می‌کنند، خراب می‌شوند. اگر یک اسکریپت برای یک دستور خطای "No such file or directory" ایجاد کند، تایپ کردن `which command` دستور مشکل‌زا است، باید به شما در پیدا کردن آن کمک کند.

نکته: استاندارد سلسله مراتب فایل لینوکس (FHS) File Hierarchy Standard برای انواع مختلف فایل‌ها تعریف می‌کند، مانند /bin برای فایل‌های اجرایی برنامه‌های محلی. بسیاری از توزیع‌های لینوکس در حال ادغام ساختار FHS در سلسله مراتب دایرکتوری خود هستند.

هر خط اجرای برنامه در لیست 11.1 به یک علامت & ختم می‌شود. این کاراکتر به شل می‌گوید که بدون منتظر ماندن برای اتمام اولین خط، به خط بعدی برود. اگر علامت‌های & را در لیستینگ 11.1 حذف کنید، نتیجه این خواهد بود که اولین xterm باز می‌شود اما دومی تا زمانی که اولی بسته نشود، باز نخواهد شد. به همین ترتیب، KMail تا زمانی که دومین xterm خاتمه نیابد، شروع نخواهد شد.

اگرچه اجرای چندین برنامه از یک اسکریپت می‌تواند در اسکریپت‌های استارت‌اپ و برخی دیگر از موقعیت‌ها در زمان صرفه‌جویی کند، اسکریپت‌ها نیز اغلب برای اجرای یک سری برنامه‌هایی که به نوعی داده‌ها را دستکاری می‌کنند، استفاده می‌شوند. چنین اسکریپت‌هایی معمولاً شامل علامت‌های & در انتهای دستورات نیستند زیرا یک دستور باید بعد از دیگری اجرا شود یا حتی ممکن است به خروجی دستور اول متکی باشد. لیست جامع از چنین دستوراتی غیرممکن است زیرا می‌توانید هر برنامه‌ای که در لینوکس نصب می‌کنید را به عنوان یک دستور در یک اسکریپت اجرا کنید. چند دستور که به طور معمول در اسکریپت‌ها استفاده می‌شوند عبارتند از:

دستورات معمولی برای دستکاری فایل: دستورات دستکاری فایل‌ها، مانند cp، mv، rm و ls، اغلب در اسکریپت‌ها استفاده می‌شوند. می‌توانید از این دستورات برای کمک به اتوماسیون وظایف تکراری نگهداری فایل‌ها استفاده کنید.

**grep**: دستور grep فایل‌هایی که شامل رشته‌ای که شما مشخص می‌کنید را پیدا می‌کند، یا خطوطی که آن رشته‌ها را در یک فایل تکی شامل می‌شوند را نمایش می‌دهد. دستور grep به تفصیل در فصل 8، "Tوضیح داده شده است. Searching, Extracting, and Archiving Data"

**find**: دستور find بر اساس الگوهای نام فایل، مالکیت و ویژگی‌های مشابه جستجو می‌کند. این دستور در فصل 8 توضیح داده شده است.

**cut**: دستور cut متن را از فایلهای یک فایل استخراج می‌کند. این دستور معمولاً برای استخراج اطلاعات متغیر از فایلی که محتوای آن به شدت الگو دار است استفاده می‌شود. برای استفاده از آن، یک یا چند آپشن که مشخص می‌کند چه اطلاعاتی می‌خواهد، و سپس یک یا چند نام فایل را وارد می‌کنید. برای مثال،

دایرکتوری‌های home کاربران در ششمين فیلد که با دونقطه جدا شده، در فایل /etc/passwd ظاهر می‌شود.  
بنابراین می‌توانید برای استخراج این اطلاعات دستور زیر را تایپ کنید

```
cut -f 6 -d ":" /etc/passwd
```

همین دستور در یک اسکریپت این اطلاعات را استخراج خواهد کرد، که احتمالاً آن را به یک متغیر ذخیره کرده  
یا به دستور بعدی منتقل می‌کند.

برنامه sed: برنامه sed بسیاری از قابلیت‌های یک تکست ادیتور معمولی (مانند عملیات جستجو و جایگزینی) را  
فراهم می‌کند، اما از طریق دستورات که می‌توانند در یک کامند لاین تایپ شده یا در یک اسکریپت وارد  
شوند.

فرمان echo: فرمان echo ابزاری است که زمانی استفاده می‌شود که یک اسکریپت باید پیغامی را به کاربر ارائه  
دهد. می‌توانید گزینه‌های مختلفی را به echo پاس بدھید یا فقط یک رشته برای نمایش به کاربر. به عنوان  
مثال، `echo "Press the Enter key" باعث می‌شود اسکریپت رشته مشخص شده را نمایش دهد. همچنین  
می‌توانید از echo برای نمایش مقدار متغیرها (که بعداً در بخش "Using Variables" توضیح داده شده)  
استفاده کنید.

فرمان mail: فرمان mail می‌تواند برای ارسال ایمیل از داخل یک اسکریپت استفاده شود. `mail -s subject` را برای  
مشخص کردن خط موضوع به آن پاس دهید و یک آدرس ایمیل را به عنوان آخرین آرگومان بدھید. اگر این  
فرمان در کامند لاین استفاده شود، سپس پیامی را تایپ کرده و با فشار دادن کلید `Ctrl+D` آن را خاتمه  
دهید. اگر از داخل یک اسکریپت استفاده شود، می‌توانید موضوع را به طور کامل حذف کنید یا یک فایل  
خارجی را به عنوان پیام با استفاده از ریدایرکشن ورودی پاس دهید. ممکن است بخواهید از این فرمان برای  
ارسال ایمیل به کاربر اصلی درباره اقدامات یک اسکریپت راهاندازی یا اسکریپتی که به صورت خودکار اجرا  
می‌شود استفاده کنید.

نکته: فصل 8 ریدایرکشن ورودی را توصیف می‌کند.

بسیاری از این دستورات بسیار پیچیده هستند و توضیح کامل آنها فراتر از محدوده این فصل است. برای  
اطلاعات بیشتر می‌توانید صفحات راهنمای (man pages) این دستورات را بررسی کنید. برخی از آنها در  
جهای دیگر این کتاب توضیح داده شده‌اند، همان‌طور که در توضیحاتشان ذکر شده است.

حتی اگر به طور کامل نحوه استفاده از برخی دستورات کلیدی خارجی را درک کنید، اجرای دستورات به همان  
صورتی که آنها را در کامند لاین تایپ می‌کنید، کاربرد محدودی دارد. بسیاری از وظایف مدیریتی نیازمند تغییر  
آنچه در یک فرمان تایپ می‌کنید یا حتی دستورات وارد شده، بر اساس اطلاعات حاصل از دستورات دیگر  
هستند. به همین دلیل، زبان‌های اسکریپتنویسی شامل ویژگی‌های اضافی هستند تا به شما کمک کنند  
اسکریپت‌های خود را مفیدتر کنید.

## Using Arguments

متغیرها می‌توانند به شما کمک کنند تا کاربرد اسکریپت‌ها را گسترش دهید. یک متغیر در اسکریپت، جایگزینی برای یک مقدار است که در زمان اجرای اسکریپت تعیین می‌شود. مقادیر متغیرها می‌توانند به عنوان پارامتر به اسکریپت منتقل شوند، به طور داخلی در اسکریپت تولید شوند، یا از محیط اسکریپت استخراج شوند. (محیط مجموعه‌ای از متغیرها است که هر برنامه‌ای می‌تواند به آن‌ها دسترسی داشته باشد. محیط شامل چیزهایی مانند دایرکتوری فعلی و مسیر جستجو برای اجرای برنامه‌ها است.)

متغیرهایی که به اسکریپت منتقل می‌شوند اغلب پارامترها یا آرگومان‌ها نامیده می‌شوند. این متغیرها در اسکریپت با علامت دلار (\$) و سپس یک عدد از 0 به بالا نشان داده می‌شوند؛ \$0 نام اسکریپت، \$1 اولین پارامتر اسکریپت، \$2 دومین پارامتر و به همین ترتیب است. برای درک چگونگی استفاده از این مفهوم، به "Creating Users and Groups" در فصل 13 توضیح داده شد، ایجاد یک حساب کاربری جدید معمولاً شامل اجرای حداقل دو دستور useradd و passwd می‌شود. ممکن است نیاز به اجرای دستورات اضافی خاص سایت نیز داشته باشد، مانند دستورات ایجاد دایرکتوری‌های خاص کاربر به جز دایرکتوری home کاربر.

به عنوان یک نمونه از اینکه چگونه یک اسکریپت با متغیر آرگومان می‌تواند در چنین موقعیت‌هایی کمک کند، لیست 11.2 را در نظر بگیرید. وقتی شما این اسکریپت را اجرا می‌کنید، باید نام اکانت را به عنوان پارامتر در کامند لاین وارد کنید. اسکریپت مقدار این پارامتر را با استفاده از متغیر \$1 به دست می‌آورد و بر اساس این مقدار با استفاده از دستور useradd یک اکانت ایجاد می‌کند. سپس با استفاده از دستور passwd رمز عبور اکانت را تغییر می‌دهد (وقتی اسکریپت را اجرا می‌کنید، اسکریپت شما را به وارد کردن رمز عبور دعوت می‌کند). این اسکریپت یک دایرکتوری در ساختار درخت دایرکتوری shared/ بر اساس حساب کاربری ایجاد می‌کند و یک لینک نمادین به آن دایرکتوری از دایرکتوری خانه جدید کاربر ایجاد می‌کند. همچنین مالکیت و دسترسی‌ها را به نحوی که ممکن است براساس سیاست‌های مالکیت و دسترسی سیستم شما مفید تنظیم می‌کند.

Listing 11.2: A script that reduces account-creation tedium

```
#!/bin/bash

useradd -m $1

passwd $1

mkdir -p /shared/$1

chown $1.users /shared/$1

chmod 775 /shared/$1

ln -s /shared/$1 /home/$1/shared

chown $1.users /home/$1/shared
```

با استفاده از لیست 11.2، شما فقط باید سه چیز را تایپ کنید: نام اسکریپت با نام کاربری مورد نظر و رمز عبور (دوبار). به عنوان مثال، اگر اسکریپت mkuser نام دارد، می‌توانید از آن به این شکل استفاده کنید:

```
# mkuser rblum
```

```
Changing password for user rblum
```

```
New password:
```

```
Retype new password:
```

```
passwd: all authentication tokens updated successfully
```

اغلب برنامه‌های اسکریپت به سکوت عمل می‌کنند مگر اینکه با مشکلاتی روبرو شوند، بنابراین تعامل (شامل تایپ رمز عبور که بر روی صفحه نمایش نشان داده نمی‌شود) ناشی از دستور passwd است. به عبارت دیگر، اسکریپت لیست 11.2 هفت خط دستور را با یک خط جایگزین می‌کند. هر یک از این خطوط از نام کاربری استفاده می‌کنند، بنابراین با اجرای این اسکریپت، شناسنامه خطاها تایپی نیز کاهش می‌یابد.

## Using Variables

نوع دیگری از متغیر که می‌توان از خروجی یک دستور تعیین کرد، همچنان با علامت دلار آغاز می‌شود، اما معمولاً نامی مانند \$Name یا \$Addr یا آن اختصاص داده می‌شود. (هنگام اختصاص مقادیر به متغیرها، علامت دلار حذف می‌شود، همانطور که بزودی نشان داده خواهد شد). سپس می‌توانید از این متغیرها در دستورات عادی استفاده کنید و به عنوان پارامترهای دستور به آنها ارسال می‌شود.

11.3 Listing را در نظر بگیرید که با کمک ابزار ping بررسی می‌کند که روتر کامپیوتر آیا فعال است یا خیر. این اسکریپت از دو متغیر استفاده می‌کند. اولین آن \$ip است که از خروجی دستور route با استفاده از دستورات tr و grep و cut استخراج می‌شود. هنگامی که مقداری به یک متغیر از خروجی یک دستور اختصاص می‌دهید، آن دستور باید در داخل نویسه‌های backtick (`) قرار گیرد که بیشتر صفحه‌کلیدها روی همان کلید هستند که تیلید (~) روی آن قرار دارد. این نویسه‌ها نقل قول‌های تکی معمولی نیستند که روی همان کلید همراه با نقل قول معمولی (`) روی بیشتر صفحه‌کلیدها قرار دارند. متغیر دوم ping\$ به سادگی به برنامه ping اشاره می‌کند. می‌توان آن را حذف کرد و استفاده‌های بعدی از \$ping را با مسیر کامل برنامه یا به سادگی با ping جایگزین کرد (با تکیه بر متغیر محیط \$PATH برای پیدا کردن برنامه). متغیرهای مانند این اغلب برای آسان‌تر کردن اصلاح اسکریپت در آینده استفاده می‌شوند. به عنوان مثال، اگر برنامه ping را جابجا کنید، تنها کافی است یک خط از اسکریپت را اصلاح کنید. متغیرها همچنین می‌توانند با شرایطی همراه شوند تا اطمینان حاصل شود که اسکریپت بر روی سیستم‌های مختلف کار می‌کند، برای مثال اگر بر روی برخی سیستم‌ها ping با نام دیگری فراخوانی شود.

Listing 11.3: Script demonstrating assignment and use of variables

```
#!/bin/bash

ip=`route -n | grep UG | tr -s " " | cut -f 2 -d " "`

ping="/bin/ping"

echo "Checking to see if $ip is up..."

$ping -c 5 $ip
```

نکته: در اضافه به چند دستور دیگر، خط `=`` از `ip` استفاده می‌کند تا خروجی زنجیره دستورات را به متغیر `ip` اختصاص دهد. فصل 8 این تکنیک را توضیح می‌دهد.

در عمل، برای استفاده از لیست 11.3 شما کافیست نام اسکریپت را تایپ کنید. نتیجه باید شامل پیام "Checking to see if 192.168.1.1 is up" با اینکه 192.168.1.1 توسط گیت‌وی پیش‌فرض سیستم جایگزین شده است و خروجی دستور `ping` باشد. این دستور باید تلاش کند تا پنج بسته را به روتر ارسال کند. اگر روتر روشن بوده و پیکربندی شده باشد تا به پینگ‌ها پاسخ دهد، شما پنج بسته بازگشتی و اطلاعات خلاصه‌ای مشابه زیر مشاهده خواهید کرد:

```
$ routercheck

Checking to see if 192.168.1.1 is up...

PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.

64 bytes from 192.168.1.1: icmp_seq=1 ttl=63 time=23.0 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=63 time=0.176 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=63 time=0.214 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=63 time=0.204 ms
64 bytes from 192.168.1.1: icmp_seq=5 ttl=63 time=0.191 ms

--- 192.168.1.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.176/4.758/23.005/9.123 ms
```

اگر روتر خاموش باشد، شما پیام‌های خط را مشاهده خواهید کرد که می‌گوید میزبان قابل دسترسی نیست.

لیست 11.3 کاربرد عملی محدودی دارد و باگ‌هایی دارد. به عنوان مثال، این اسکریپت تنها با حضور رشته `UG` در خط خروجی روتر از دستور `route`، در واقعیت گیت‌وی کامپیوتر را تعیین می‌کند. اگر یک کامپیوتر دو روتر تعریف شده داشته باشد، این روش به درستی کار نمی‌کند و نتیجه این خواهد بود که اسکریپت

ناخواسته عمل می‌کند. هدف از لیست 11.3 نشان دادن این است که چگونه می‌توان متغیرها را اختصاص داده و استفاده کرد، نه اینکه یک اسکریپت کارآمد و بی‌نقص باشد.

اسکریپتهای مانند لیست 11.3 که اطلاعات را از اجرای یک یا چند دستور دریافت می‌کنند، مفید برای پیکربندی ویژگی‌هایی هستند که بسته به اطلاعات خاص سیستم یا اطلاعاتی که با زمان تغییر می‌کنند، متغیر هستند. شما می‌توانید از این روش مشابه برای دریافت نام کامپیوتر فعلی (استفاده از دستور hostname)، زمان فعلی (استفاده از دستور date)، مجموع زمانی که کامپیوتر روشن بوده است (استفاده از دستور uptime)، فضای آزاد دیسک (استفاده از دستور df) و غیره استفاده کنید. هنگامی که با عبارات شرطی (که به زودی شرح داده می‌شود) ترکیب می‌شوند، متغیرها قدرت بیشتری پیدا می‌کنند، زیرا اسکریپت شما می‌تواند یک عملیات را انجام دهد وقتی که یک شرایط برقرار است و در شرایط دیگر دیگر را انجام دهد. به عنوان مثال، یک اسکریپت که نرم‌افزار را نصب می‌کند می‌تواند فضای آزاد دیسک را بررسی کند و اگر فضای کافی موجود نبود، نصب را لغو کند.

علاوه بر اختصاص متغیرها با اپراتور اختصاص (=)، شما می‌توانید متغیرها را از ورودی استاندارد با استفاده از دستور read خوانده و آن‌ها را به صورت \$response برای دسترسی بعدی می‌سازید. این روش اختصاص متغیر برای اسکریپتهایی کاربرد دارد که باید با کاربران تعامل داشته باشند. به عنوان مثال، به جای خواندن نام کاربری از کامند لاین، لیست 11.2 می‌تواند به‌گونه‌ای تغییر یابد که از کاربر نام کاربری را بپرسد. لیست 11.4 نتیجه را نشان می‌دهد. برای استفاده از این اسکریپت، شما نام آن را بدون تایپ کردن نام کاربری در کامند لاین می‌نویسید. سپس اسکریپت از شما برای نام کاربری درخواست می‌دهد و پس از وارد کردن آن، تلاش می‌کند که یک حساب با آن نام ایجاد کند.

Listing 11.4: Modified version of Listing 11.2 that employs user interaction

```
#!/bin/bash
echo -n "Enter a username: "
read name
useradd -m $name
passwd $name
mkdir -p /shared/$name
chown $name.users /shared/$name
chmod 775 /shared/$name
ln -s /shared/$name /home/$name/shared
chown $name.users /home/$name/shared
```

یک نوع ویژه از متغیر، متغیر محیطی است که به آن یک متغیر اسکریپت شل اختصاص داده شده و دسترسی به آن را به همان شکل دارید. تفاوت آن این است که اسکریپت یا دستوری که یک متغیر محیطی را

تنظیم می‌کند، از دستور `export Bash` استفاده می‌کند تا ارزش متغیر قابل دسترسی برای برنامه‌هایی که از شل یا اسکریپت شلی که اختصاص داده شده است، را فراهم آورد. به عبارت دیگر، شما می‌توانید یک متغیر محیطی را در یک اسکریپت تنظیم کنید و در یک اسکریپت دیگری که توسط اسکریپت اول راهاندازی شده است، از آن استفاده کنید. معمولاً متغیرهای محیطی در اسکریپتهای راهاندازی شل تنظیم می‌شوند، اما اسکریپتهای شما می‌توانند به آنها دسترسی پیدا کنند. به عنوان مثال، اگر اسکریپت شما برنامه‌های X را فراخواند، ممکن است بررسی کند که آیا متغیر محیطی `$DISPLAY` معتبر است یا نه و در صورت عدم تنظیم این متغیر، عملیات را متوقف کند. طبقه‌بندی متغیرهای محیطی به صورت کلی حروف بزرگ هستند، در حالی که متغیرهای اسکریپت شلی غیرمحیطی معمولاً حروف کوچک یا مخلوط هستند.

یک متغیر ویژه ارزش ذکر دارد: `$?` این متغیر مقدار خروجی (یا `return value`) از آخرین دستور اجرا شده را نگه می‌دارد. خروجی اینتر، یک مقدار عددی است که می‌توانید آن را بررسی کنید تا تصمیم بگیرید که آیا دستور به درستی اجرا شده است یا نه. اکثر برنامه‌ها مقدار 0 را بر می‌گردانند وقتی به طور نرم‌الپایان یافته‌اند و مقدار دیگری را برای مشخص کردن خطاهای بر می‌گردانند. شما می‌توانید این مقدار را با دستور `echo` نمایش دهید یا از آن در یک عبارت شرطی (که در زیر توضیح داده شده است) استفاده کنید تا اسکریپت شما دستورات خاص برای مدیریت خطای اجرا کند.

**نکته:** برای دریافت معانی مقادیر بازگشته یک برنامه، می‌توانید به صفحه راهنمای آن برنامه (`man page`) مراجعه کنید.

## Using Conditional Expressions

زبان‌های اسکریپت نویسی از چندین نوع عبارت شرطی پشتیبانی می‌کنند. این‌ها به اسکریپت این امکان را می‌دهند که به وابستگی به شرایط خاص، یکی از چندین عملیات را اجرا کند - معمولاً بر اساس مقدار یک متغیر. یکی از دستورات شایع که از عبارات شرطی استفاده می‌کند، دستور `if` است که به سیستم اجازه می‌دهد بر اساس اینکه یک شرط صحیح باشد یا خیر، یکی از دو عملیات را انجام دهد. عبارت شرطی برای دستور `if` در پرانتزهای درونی پس از کلمه `if` ظاهر می‌شود و می‌تواند اشکال مختلفی داشته باشد. عبارت‌های شرطی از گزینه‌ها و عملگرها برای تعریف شرایط استفاده می‌کنند. به عنوان مثال، شرط زیر:

```
[ -f
file ]
```

از گزینه `f`-استفاده می‌کند و درست است اگر فایل وجود داشته باشد و فایل منظم باشد؛ در حالی که شرط:

```
[ -s
file ]
```

از گزینه `s`-استفاده می‌کند و درست است اگر فایل وجود داشته باشد و اندازه‌ی فایل بیشتر از صفر باشد. همچنین می‌توان از برخی عملگرها رشته‌ای در شرایط استفاده کرد. به عنوان مثال:

```
[string1 == string2 ]
```

از عملگر == استفاده می‌کند و درست است اگر دو رشته‌ی مقدار یکسانی داشته باشند. برای بهتر فهمیدن استفاده از شرایط، قطعه کد زیر را در نظر بگیرید:

```
if [ -s /tmp/tempstuff ]
then
    echo "/tmp/tempstuff found; aborting!"
    exit
fi
```

این قطعه کد باعث می‌شود که اسکریپت در صورت وجود فایل /tmp/tempstuff / خاتمه یابد. واژه then نشان‌دهنده آغاز خطوطی است که فقط در صورت درست بودن شرط اجرا می‌شوند، و واژه fi (بر عکس if) نشان‌دهنده پایان بلوک if است. این کد ممکن است مفید باشد اگر اسکریپت این فایل را ایجاد و در آخر حذف کند، زیرا حضور آن نشان‌دهنده‌ی این است که یک اجرای قبلی از اسکریپت موفق نبوده است یا هنوز ادامه دارد.

یک فرم جایگزین برای یک عبارت شرطی استفاده از واژه test است به جای کروشهای مربع اطراف شرطی است:

```
if test -s /tmp/tempstuff
```

می‌توانید همچنین مقدار برگشتی یک دستور را با استفاده از خود دستور به عنوان شرط بررسی کنید:

```
if [ command ]
then
    additional-commands
fi
```

در این مثال، دستورات اضافی (`additional-commands`) فقط در صورتی اجرا می‌شوند که دستور اصلی (`command`) با موفقیت اجرا شود. اگر دستور اصلی یک کد خطا برمی‌گرداند، دستورات اضافی اجرا نمی‌شوند.

عبارت‌های شرطی می‌توانند با استفاده از بخش else گسترش یابند:

```

if [ conditional-expression ]
then
  commands
else
  other-commands
fi

```

کدی به این شکل باعث می‌شود که نسبت به ارزیابی conditional-expression یا other-commands ر اجرا شوند. این حالت زمانی مفید است که چیزی باید در بخشی از برنامه رخ دهد، اما دقیقاً چه چیزی باید رخ دهد به یک شرط بستگی دارد. به عنوان مثال، ممکن است بخواهید یکی از دو برنامه آرشیو فایل مختلف را بسته به ورودی کاربر راهاندازی کنید.

اگر بیش از دو نتیجه ممکن باشد، برای مثال، اگر کاربر ممکن است یکی از چهار ورودی ممکن را ارائه دهد، چه کاری انجام می‌دهید؟ شما می‌توانید چندین عبارت if/then/else را تودرتو کنید، اما این کار سریع نامرتب می‌شود. یک رویکرد تمیزتر استفاده از case است:

```

case word in
  pattern1) command(s) ;;
  pattern2) command(s) ;;
  ...
esac

```

برای یک عبارت word، یک احتمالاً یک متغیر است و هر pattern یک مقدار ممکن برای آن متغیر است. الگوهای می‌توانند مشابه نام فایل‌ها گسترش یابند، با استفاده از همان کاراکترهای جایگزین و قوانین گسترش (به عنوان مثال، \* برای نشان دادن هر رشته). شما می‌توانید به تعداد دلخواه الگوها را به این روش مطابقت دهید. هر مجموعه از دستورات باید با یک نقطه‌ویرگول دوبل (;;) خاتمه یابد و عبارت case به طور کلی با رشته esac (برعکس case) پایان می‌یابد.

نکته: گسترش نام فایل با استفاده از ستاره‌ها (\*)، علامت‌های سوال (?) و غیره، گاهی اوقات globbing نامیده می‌شود.

هنگام اجرا، bash دستورات مرتبط با اولین الگوی تطبیق داده شده با کلمه را اجرا می‌کند. سپس اجرا به خط پس از عبارت esac می‌پردازد؛ هر دستوری که در این میان قرار گرفته باشد اجرا نمی‌شود. اگر هیچ الگویی با کلمه مطابقت نداشته باشد، هیچ کدی درون عبارت case اجرا نمی‌شود. اگر می‌خواهید یک شرط پیش‌فرض

داشته باشید، از \* به عنوان الگوی نهایی استفاده کنید؛ این الگو با هر کلمه‌ای مطابقت دارد، بنابراین دستورات آن اجرا می‌شوند اگر هیچ الگوی دیگری مطابقت نداشته باشد.

## Using Loops

عبارات شرطی گاهی در حلقه‌ها استفاده می‌شوند. حلقه‌ها ساختارهایی هستند که به اسکریپت می‌گویند یک کار را به طور مکرر انجام دهد تا زمانی که یک شرط برآورده شود (یا تا زمانی که یک شرط دیگر برآورده نشود). به عنوان مثال، فهرست 11.5 یک حلقه را نشان می‌دهد که همه فایل‌های صوتی WAV در یک پوشه را پخش می‌کند.

نکته: دستور aplay یک پخش‌کننده ساده فایل‌های صوتی است. در برخی سیستم‌ها، ممکن است به جای aplay نیاز به استفاده از play یا دستور دیگری داشته باشید.

Listing 11.5: A script that executes a command on every matching file in a directory

```
#!/bin/bash

for d in `ls *.wav` ; do
    aplay $d
done
```

حلقه for به این شکل برای هر مورد در لیستی که توسط دستور ls \*.wav تولید می‌شود، یک بار اجرا می‌شود. هر یک از این موارد (نام فایل‌ها) به نوبت به متغیر \$d اختصاص داده شده و سپس به دستور aplay ارسال می‌شود.

دستور seq می‌تواند در ایجاد حلقه‌های for (و در موارد دیگر نیز) مفید باشد. این دستور لیستی از اعداد را از آرگومان اول خود شروع کرده و تا آرگومان آخر خود ادامه می‌دهد. برای مثال، تایپ کردن seq 1 10 ده خط تولید می‌کند که هر خط شامل عددی بین 1 تا 10 است. شما می‌توانید از دستور seq در یک حلقه for برای پیمایش از طریق یک سری اعداد استفاده کنید:

```
for x in `seq 1 10` ; do
    echo $x
done
```

این حلقه 10 بار اجرا می‌شود و مقدار x در هر تکرار افزایش می‌یابد. اگر تنها یک پارامتر به دستور seq بدهید، آن را به عنوان نقطه پایان تفسیر می‌کند و نقطه شروع را 1 در نظر می‌گیرد. اگر سه مقدار به seq بدهید، آنها را به عنوان مقدار شروع، مقدار افزایش، و مقدار پایان تفسیر می‌کند.

نوع دیگری از حلقه حلقه while است که تا زمانی که شرط آن درست باشد اجرا می‌شود. شکل اصلی این نوع حلقه به صورت زیر است:

```
while [ condition ]
```

```
do
```

```
    commands
```

```
done
```

حلقه until در ساختار مشابهی است، اما تا زمانی که شرط آن نادرست باشد (یعنی تا زمانی که شرط درست شود) ادامه می‌یابد.

## Using Functions

تابع یک بخش از یک اسکریپت است که یک زیر وظیفه خاص را انجام می‌دهد و می‌توان آن را با نام آن از بخش‌های دیگر اسکریپت صدا زد. توابع با قرار دادن پرانتزها پس از نام تابع و درون آن با استفاده از پرانتزهای زائد مربع محدود می‌شوند:

```
myfn() {
```

```
    commands
```

```
}
```

کلمه کلیدی function ممکن است به صورت اختیاری قبل از نام تابع قرار گیرد. در هر حالتی، تابع با صدا زدن نام خود به عنوان یک دستور داخلی یا خارجی مانند دستورات معمول فراخوانی می‌شود.

توابع بسیار مفید هستند تا به ایجاد اسکریپتهای مازولار کمک کنند. به عنوان مثال، اگر اسکریپت شما نیاز به انجام شش محاسبه متمایز دارد، می‌توانید هر محاسبه را در یک تابع قرار دهید و سپس همه آن‌ها را به ترتیب فراخوانی کنید. لیست 11.6 نحوه استفاده از توابع را در یک برنامه ساده نشان می‌دهد که یک فایل را کپی می‌کند اما در صورت وجود فایل مقصد، با یک پیام خطای متوقف می‌شود. این اسکریپت نام فایل مقصد و مقصد را پذیرفته و باید این نام‌های فایل را به توابع منتقل کند.

Listing 11.6: A script demonstrating the use of functions

```
#!/bin/bash

doit() {

    cp $1 $2

}

function check() {

    if [ -s $2 ]

        then

            echo "Target file exists! Exiting!"

            exit

    fi

}

check $1 $2

doit $1 $2
```

اگر شما لیست 11.6 را وارد کنید و آن را safercp نام‌گذاری کنید، می‌توانید آن را به این صورت استفاده کنید، با فرض اینکه فایل original.txt وجود دارد و فایل dest.txt وجود ندارد:

```
$ ./safercp original.txt dest.txt
```

```
$ ./safercp original.txt dest.txt
```

Target file exists! Exiting!

اولین اجرای اسکریپت با موفقیت انجام شد زیرا فایل dest.txt وجود نداشت. در اجرای دوم، فایل مقصد وجود داشته، بنابراین اسکریپت با پیام خطا خاتمه یافت.

توجه داشته باشید که توابع به طور مستقیم و به ترتیبی که در اسکریپت ظاهر شده‌اند اجرا نمی‌شوند. آن‌ها فقط زمانی اجرا می‌شوند که در بدنه اصلی اسکریپت فراخوانده شوند. در Listing 11.6، این بدنه اصلی اسکریپت فقط دو خط است که هرکدام به یک فراخوانی تابع مرتبط است و در انتهای اسکریپت قرار دارند.



## Administrator Shell Scripts

کار یک مدیر لینوکس در برخی مواقع می‌تواند کمی خسته‌کننده باشد. معمولاً مدیران باید روزانه بسیاری از فایل‌های log را بررسی کنند تا اطمینان حاصل کنند که همه چیز به درستی کار می‌کند و هیچ نقص امنیتی در سیستم رخ نداده است. این کار ممکن است زمان زیادی را به خود اختصاص دهد.

اما بسیاری از مدیران لینوکس از اسکریپت‌های شل استفاده می‌کنند تا کمک کنند تا بخشی از خستگی مربوط به بررسی صفحات فایل‌های log را از بین ببرند. به جای جستجوی دستی در فایل‌های log، مدیران اسکریپت‌هایی می‌نویسند که از ابزارهای پردازش متنی لینوکس مانند grep و cut برای جستجوی خطاهای هشدارها در فایل‌های log استفاده می‌کنند، سپس این خطاهای هشدارها را به یک فایل متنی جداگانه کپی می‌کنند و آن را به خودشان ایمیل می‌کنند. سپس این اسکریپت‌ها را برنامه‌ریزی می‌کنند تا هر شب اجرا شوند، به طوری که صبح اولین کاری که وقتی به محل کار می‌رسند، داشته باشند یک خلاصه از هر مشکل سیستمی که وجود داشته است. این به آن‌ها امکان می‌دهد تا انرژی خود را برای حل مشکلات سیستم به کار بگیرند به جای این‌که فقط به دنبال آن‌ها بگردند!

## Setting the Script's Exit Value

معمولًا، وضعیت خروجی یک اسکریپت مانند وضعیت آخرین دستوری است که اجرا شده است، به این معنی که اسکریپت مقدار \$? را بر می‌گرداند. با این حال، شما می‌توانید وضعیت خروجی را کنترل کنید و یا در هر نقطه‌ای از اسکریپت با استفاده از دستور exit خروج کنید. استفاده از دستور exit بدون هیچ گزینه‌ای، منجر به قطع فوری اجرا اسکریپت با وضعیت خروجی معمولی \$? می‌شود. این می‌تواند در اداره خطاهای یا لغو یک عملیات در حین اجرا به دلایل مختلف مفید باشد؛ اگر اسکریپت خط را شناسایی کند یا اگر کاربر گزینه‌ای برای خاتمه دادن انتخاب کند، می‌توانید از exit برای خاتمه دادن به اسکریپت استفاده کنید.

اگر یک مقدار عددی بین 0 تا 255 به دستور exit بدهید، اسکریپت قطع شده و مقدار مشخص شده را به عنوان خروجی خود اسکریپت باز می‌گرداند. شما می‌توانید از این ویژگی برای ارسال خطاهای به اسکریپت‌های دیگری که ممکن است اسکریپت شما را فراخوانی کنند، استفاده کنید. با این حال، ممکن است نیاز به اضافه کردن کد اضافی برای پیگیری علل پایان ناگهانی داشته باشید. به عنوان مثال، می‌توانید یک متغیر (به عنوان مثال \$termcause) را برای نگه‌داری علت پایان اسکریپت تعریف کنید. این متغیر را در ابتدای اسکریپت به 0 تنظیم کنید و اگر اسکریپت مشکلی را شناسایی کند که باعث قطع شدن اسکریپت می‌شود، را به یک مقدار غیر از 0 تغییر دهید. (می‌توانید از هر کد عددی که دوست دارید استفاده کنید؛ معنای مشخصی برای این کدها وجود ندارد.) در هنگام خروج، از exit مطمئن شوید که را به عنوان آرگومان ارسال کنید:

```
exit $termcause
```

# CHAPTER 12

## Understanding Basic Security

لینوکس یک سیستم عامل چند کاربره است، به این معنی که ویژگی‌هایی را فراهم می‌کند که به چندین فرد امکان استفاده از کامپیوتر را می‌دهد. به طور کلی، این ویژگی‌ها اکانت‌ها را تشکیل می‌دهند. فصل‌های قبلی این کتاب به طور مختصر به حساب‌ها اشاره کرده‌اند، اما به تفصیل به آنها پرداخته نشده است.

این فصل این موضوع را تغییر می‌دهد؛ اصول مهم اکانت و چند دستور که می‌توانید برای بررسی اکانت استفاده کنید را شرح می‌دهد. گروه‌ها نیز به اکانت‌ها مرتبط هستند که مجموعه‌ای از اکانت‌ها هستند که می‌توانند دسترسی‌های ویژه‌ای به کامپیوتر داشته باشند، بنابراین این فصل همچنین درباره گروه‌ها توضیح می‌دهد. یک اکانت به نام `root` دارای امتیازات ویژه‌ای بر روی کامپیوتر است. برخی مدیران از این اکانت برای انجام وظایف مدیریت سیستم استفاده می‌کنند، اما اینکه این روش به عنوان یک شیوه بد مدیریت شود در نظر گرفته می‌شود. بنابراین، قبل از آنکه به وظایف مدیریتی که در چند فصل آخر این کتاب توضیح داده شده است، بپردازید، باید این اکانت را درک کنید و بدانید چه چیزی باید به جای آن استفاده کنید. این موضوعات مهم در بنیان امنیت سیستم قرار دارند.

## Understanding Accounts

اکانت‌ها به کاربران امکان می‌دهند که بتوانند از یک کامپیوتر مشترک استفاده کنند بدون اینکه مشکلات زیادی برای یکدیگر ایجاد کنند. همچنین به مدیران سیستم این امکان را می‌دهد که بررسی کنند که چه کسانی از منابع سیستم استفاده می‌کنند و گاهی هم کسانی که کارهایی را انجام می‌دهند که نباید انجام دهنند را شناسایی کنند. ویژگی‌های اکانت به کاربران کمک می‌کند تا از کامپیوتر استفاده کنند و به مدیران کمک می‌کند تا آن را مدیریت کنند. درک این ویژگی‌ها پایه‌ای برای مدیریت حساب‌ها است.

نکته: حتی یک کامپیوتر کاربردی با یک اکانت نیز از چند اکانت سیستمی استفاده می‌کند. این نوع کامپیوتر ممکن است فقط یک اکانت کاربری داشته باشد، اما چندین اکانت سیستمی نیز دارد که به کمک اجرای صحیح کامپیوتر کمک می‌کنند.

بعضی از ویژگی‌های حساب کمک می‌کند تا شما اکانت‌ها را شناسایی کنید و فایل‌ها و منابع مرتبط با آن‌ها را پیدا کنید. دانستن چگونگی استفاده از این ویژگی‌ها به شما کمک می‌کند تا مشکلات مرتبط با اکانت‌ها را پیدا کرده و کاربران کامپیوتر را مدیریت کنید.

نکته: توضیحاتی درباره انواع مختلف حساب‌ها وجود دارد – اکانت‌های عمومی، اکانت‌های سیستمی و اکانت `root`. اکانت عمومی برای کاربرانی طراحی شده‌اند که نیازی به دسترسی ویژه برای انجام کارهای روزمره‌شان ندارند، مانند ایجاد اسناد پردازش متن. اکانت سیستمی برای خدمات و برنامه‌های خاصی نظیر سرویس‌های ارائه‌دهنده صفحات وب راهاندازی می‌شوند. اکانت `root` به طور تاریخی (و گاهی هم اکنون) برای انجام وظایف

مدیریت سیستم استفاده می‌شده است. این مباحث به طور دقیق‌تر در بخش "Understanding User Types" در ادامه این فصل بررسی خواهد شد.

## Understanding Account Features

اکثر ویژگی‌های اکانت‌ها در فایل /etc/passwd تعریف شده‌اند که از خطوطی با جداگانه دو نقطه تشکیل شده است. هر خط (یا رکورد) یک اکانت را تعریف می‌کند. یک ورودی ممکن است به شکل زیر باشد:

```
rich:x:1001:1001:Richard Blum:/home/rich:/bin/bash
```

اطلاعات موجود در فیلدهای این رکورد شامل موارد زیر است:

**Username:** نام کاربری یک اکانت مهم‌ترین ویژگی آن است. بیشتر نام‌های کاربری لینوکس شامل حروف کوچک و گاهی اعداد هستند، مانند rich یا 1138thx. کاراکترهای زیرخط (\_) و خط تیره (-) نیز در برخی توزیع‌های لینوکس در نام‌های کاربری معتبر هستند.

**Password:** اکانت‌ها معمولاً با رمز عبور محافظت می‌شوند که برای ورود به سیستم مورد نیاز است. ورود مستقیم به اکثر اکانت‌های سیستمی غیرفعال است، بنابراین آن‌ها رمز عبور ندارند. (حساب کاربری root در برخی توزیع‌ها استثنای مهمی است؛ ممکن است رمز عبور داشته باشد.) فیلد رمز عبور در فایل /etc/passwd معمولاً حاوی X است که کدی به معنی ذخیره شدن رمز عبور در /etc/shadow است، که در ادامه توضیح داده خواهد شد.

**UID:** در واقعیت، نام کاربری فقط یک برچسب است که کامپیوتر به ما انسان‌های عددنշناس نمایش می‌دهد. کامپیوتر از شماره شناسایی کاربر (UID) برای پیگیری اکانت‌ها استفاده می‌کند. شماره‌های UID با 0 شروع می‌شوند (که به حساب root اشاره دارد). در بیشتر توزیع‌ها، اکانت‌ها دارای شماره‌های UID از 1000 به بالا هستند و شماره‌های پایین‌تر برای اکانت‌های سیستمی محفوظ هستند.

نکته: برخی توزیع‌ها اکانت‌ها را از شماره 500 به جای 1000 شماره‌گذاری می‌کنند.

**GID:** اکانت‌ها به یک یا چند گروه مرتبط هستند که از بسیاری جهات مشابه اکانت‌ها هستند؛ با این حال، یک گروه مجموعه‌ای از اکانت‌های است. یکی از اهداف اصلی گروه‌ها این است که کاربران بتوانند به برخی از کاربران دیگر دسترسی به فایل‌های خود را بدهند، در حالی که کاربران غیرعضو را از دسترسی به آن‌ها بازدارند. هر اکانت مستقیماً به یک گروه اصلی از طریق یک شماره شناسایی گروه (GID) مرتبط است (100 در مثال پیشین). با قرار دادن یک اکانت در تعریف یک گروه، اکانت‌ها می‌توانند به چندین گروه مرتبط شوند همانطور که در فصل 13، "Creating Users and Groups" توضیح داده شده است.

نکته: مالکیت فایل و مجوزها در فصل 14، "Setting Ownership and Permissions" توضیح داده شده است.

**Comment Field**: فیلد توضیحات معمولاً شامل نام کامل کاربر (در این مثال Richard Blum) است، اگرچه این فیلد می‌تواند اطلاعات دیگری به جای یا علاوه بر نام کاربر داشته باشد.

**Home Directory**: اکانت‌ها و برحی از اکانت‌های سیستمی دارای دایرکتوری خانه هستند (در /home/rich در این مثال). دایرکتوری خانه محل اصلی حساب است. معمولاً مالکیت دایرکتوری خانه به حساب مربوطه تعلق دارد. ابزارها و روش‌های خاصی وجود دارند که دسترسی کاربران به دایرکتوری خانه‌شان را آسان می‌کنند؛ به عنوان مثال، علامت تیلدا (~) هنگامی که در ابتدای نام فایل استفاده شود به دایرکتوری خانه کاربر اشاره دارد.

**Default Shell**: یک شل پیشفرض به هر حسابی مرتبط است. در لینوکس، این شل معمولاً Bash است، اما کاربران می‌توانند این را به دلخواه تغییر دهند. اکثر اکانت‌های سیستمی غیر روت شل پیشفرض را به /sbin/nologin (یا /usr/sbin/nologin) تنظیم می‌کنند به عنوان یک اقدام امنیتی اضافی—این برنامه پیام نمایش می‌دهد که اکانت در دسترس نیست. استفاده از /bin/false نیز به شیوه‌ای مشابه عمل می‌کند، اگرچه بدون پیام توضیحی.

شما ممکن است از نام آن حدس بزنید که /etc/passwd اطلاعات رمز عبور را نگه می‌دارد. این معمولاً امروز اینطور نیست، اگرچه بسیاری از سال‌ها پیش چنین بود. به دلایل تاریخی، /etc/passwd باید توسط تمام کاربران قابل خواندن باشد، بنابراین ذخیره رمزهای عبور در آنجا، حتی به صورت رمز عبور Salted و هش‌شده، ریسکی است.

**نکته**: رمزهای عبور با استفاده از یک هش Salted ذخیره می‌شوند، که یک فرآیند ریاضی یک‌طرفه با ورودی تصادفی اضافی (نمک) است و چیزی تولید می‌کند که به نظر انسان‌ها بی‌معنی می‌آید. وقتی کاربر یک رمز عبور وارد می‌کند، آن رمز عبور نمکدار و هش می‌شود و اگر هش‌های نمکدار مطابقت داشته باشند، دسترسی داده می‌شود.

رمزهای عبور امروز در فایل دیگری به نام /etc/shadow ذخیره می‌شوند که کاربران عادی نمی‌توانند آن را بخوانند. این فایل یک رمز عبور نمکدار و هش شده، و همچنین اطلاعات دیگری را با یک اکانت مرتبط می‌کند. این اطلاعات می‌تواند یک اکانت را پس از یک دوره زمانی غیرفعال کند یا اگر کاربر در یک دوره زمانی مشخص رمز عبور را تغییر ندهد. یک ورودی نمونه از /etc/shadow به این شکل است:

```
rich:$6$E/moFkeT5UnTQ3KqZUoA4F12tPUoIc[...]:18114:5:30:14:-1:-1:
```

معنای هر فیلد جداشده با نقطه ویرگول در این خط به شرح زیر است:

**Username**: هر خط با نام کاربری شروع می‌شود. توجه کنید که شناسه یونیک کاربر (UID) در فایل /etc/shadow استفاده نمی‌شود؛ نام کاربری ارتباط این ورودی‌ها را با آن‌هایی که در فایل /etc/passwd هستند برقرار می‌کند.

**Password**: رمز عبور به صورت هش نمودهای ذخیره می‌شود، بنابراین شباهت ظاهری آن با رمز عبور واقعی وجود ندارد. استفاده از علامت ستاره (\*) یا علامت تعجب (!) نشانگر یک حساب با عدم وجود رمز عبور است

(یعنی حساب ورود را نمی‌بزیرد - قفل شده است). این موضوع برای حسابهای استفاده شده توسط سیستم خودش بسیار معمول است.

**Last Password Change**: فیلد بعدی (به عنوان مثال 18114 در اینجا) تاریخ آخرین تغییر رمز عبور است. این تاریخ به عنوان تعداد روزهای گذشته از 1 ژانویه 1970 ذخیره می‌شود.

نکته: زمان Epoch یا یونیکس که همچنین به آن POSIX زمان می‌گویند، تعداد ثانیه‌هایی است که از 1 ژانویه 1970 گذشته است، اگرچه فایل /etc/shadow آن را به صورت روزها نمایش می‌دهد. این مفهوم تاریخچه طولانی با سیستم‌های Unix و Linux دارد. شما نیازی ندارید که ماشین حساب خود را بیرون بکشید تا تاریخ یک فیلد را با استفاده از زمان Epoch مشخص کنید. به جای این کار، ابزار chage این کار را برای شما انجام می‌دهد، با نمایش رکورد فایل /etc/shadow برای یک اکانت مشخص به یک فرمت انسان‌پسند. زمان Epoch ممکن است در سال 2038 در سیستم‌های کوچکی که هنوز از پردازنده‌های 32 بیتی استفاده می‌کنند، مشکل ایجاد کند، زیرا کامپیوتر قادر به پیگیری زمان به درستی نخواهد بود.

**Days Until a Change Is Allowed**: فیلد بعدی (5 در این مثال) تعداد روزها قبل از اینکه امکان تغییر رمز عبور امکان‌پذیر شود را نمایش می‌دهد. این برای جلوگیری از این استفاده می‌شود که کاربران رمز عبور خود را تغییر دهند (همانطور که لازم است) و سپس آن را به رمز عبور اصلی بازگردانند.

**Days Before a Change Is Required**: این فیلد (30 در این مثال) تعداد روزها را نمایش می‌دهد که از زمان آخرین تغییر رمز عبور گذشته است، تا اینکه نیاز به تغییر دوباره آن باشد.

**Days of Warning Before Password Expiration**: اگر سیستم شما به انقضای رمز عبور تنظیم شده باشد، می‌توانید تنظیم کنید که به کاربر هشدار دهد که تاریخ انقضای نزدیک است. معمولاً مقدار 7 استفاده می‌شود. با این حال، ممکن است استفاده از 14 روز، همانطور که در مثال قبلی نشان داده شده است، مناسب باشد اگر کارمندان شرکت شما تعطیلات دو هفته‌ای داشته باشند.

**Days Between Expiration and Deactivation**: لینوکس به انتهای زمانی که اکانت منقضی می‌شود و وقتی که به طور کامل غیرفعال می‌شود، اجازه می‌دهد. یک اکانت منقضی ممکن است یا نتواند استفاده شود یا نیاز به تغییر رمز عبور فوری بعد از ورود کاربر داشته باشد. در هر دو حالت، رمز عبور آن باقی می‌ماند. رمز عبور یک اکانت غیرفعال پاک می‌شود و اکانت نمی‌تواند تا زمانی که مدیر سیستم آن را فعال کند، استفاده شود. A-1 در این فیلد، همانطور که در مثال قبلی نشان داده شده است، نشان می‌دهد که این ویژگی غیرفعال شده است.

**Expiration Date**: این فیلد تاریخی را نشان می‌دهد که حساب کاربری در آن منقضی خواهد شد. همانطور که تاریخ آخرین تغییر رمز عبور نشان داده می‌شود، تاریخ به عنوان تعداد روزهایی از 1 ژانویه 1970 به نمایش در آمده است. A-1 در این فیلد، همانطور که در مثال قبلی نشان داده شده است، نشان می‌دهد که این ویژگی غیرفعال شده است.

**Special Flag**: این فیلد برای استفاده در آینده محفوظ شده است و به طور عمومی استفاده نمی‌شود یا مقداری بی‌معنی دارد. این فیلد در مثال قبلی خالی است. برای فیلدهای مربوط به تعداد روزها، معمولاً مقداری از 1- یا 99999 یا هیچ مقدار (خالی) نشان می‌دهد که ویژگی مربوطه غیرفعال شده است. ارزش‌های /etc/shadow usermod که در فصل 13 شرح داده شده است) و chage تغییر یابند. درک قالب فایل به شما این امکان را می‌دهد که محتویات آن را بررسی کرده و هرگونه اختلاف را که ممکن است نشان دهد که سیستم شما مورد نفوذ قرار گرفته است، توجه کنید.

نکته: اصطلاحات hashed و encrypted اغلب وقتی با اشیاء کامپیوتري استفاده می‌شوند، با هم اشتباه می‌شوند. شما می‌توانید یک شیء رمزنگاری شده را رمزگشایی (decrypt) کنید، اما نمی‌توانید یک شیء هش (hash) شده را "dehash" کنید. رمز عبورها در لینوکس هش و سالت (salted and hashed) می‌شوند، اگرچه اغلب در مستندات لینوکس، اشتباهاً از اصطلاح رمزگذاری (encrypted) به جای هش استفاده می‌شود.

فایل /etc/shadow معمولاً با مجوزهای محدود و توسط کاربر root مالکیت دارد. این واقعیت حیاتی برای کارایی سیستم shadow password است زیرا این اقدام مانع از خواندن فایل و استخراج لیست رمز عبور /etc/passwd توسط کاربران غیر ادمینیستراتور می‌شود، حتی در قالب سالت و هش شده. به عکس از این، باید توسط کاربران عادی قابل خواندن باشد و معمولاً دارای مجوزهای کمتر محدودیتی است.

مهم است که درک کنید که یک اکانت همچون یک موجود تکی مانند یک فایل باینری برنامه نیست. اطلاعات اکانت در سراسر چندین فایل پیکربندی پخش شده‌اند، از جمله /etc/shadow و /etc/passwd و /etc/group و ممکن است در فایل‌های پیکربندی دیگری که به اکانت‌ها ارجاع می‌دهند. فایل‌های کاربر در دایرکتوری خانه کاربر و شاید در جاهای دیگر نیز قرار دارند. بنابراین، مدیریت اکانت‌ها ممکن است نیازمند انجام بیشتر از حفظ یک یا دو فایل باشد. به همین دلیل، ابزارها و ابزارهای کاربردی مختلف وجود دارند که به شما در ایجاد، مدیریت و حذف اکانت‌ها کمک می‌کنند، همانطور که در بخش بقیه این فصل و فصل 13 توضیح داده شده است.

نکته: بعضی از فایل‌های کاربر که خارج از دایرکتوری خانه‌ی کاربر ذخیره می‌شوند، ممکن است شامل ایمیل‌ها در /var/spool/mail و فایل‌های موقت در /tmp باشند.

## Identifying Accounts

یک راه برای شناسایی اکانت‌ها استفاده از ابزار GUI برای مدیریت اکانت است. این ابزارها از یک توزیع به دیگر متفاوتند. یک مثال از این ابزار، ابزار حساب‌ها و گروه‌ها در یک سیستم لینوکس منت است. شما می‌توانید از طریق کلیک بر روی منو در پنجره اصلی و سپس تایپ user در جعبه جستجو به این ابزار دسترسی پیدا کنید، همان‌طور که در شکل 12.1 نشان داده شده است.



Figure 12.1 Locating the Users and Groups account tool on Linux Mint

دسترسی به این گزینه یک پنجره مشابه با تصویر 12.2 را نمایش می‌دهد، اما فقط اگر شما دسترسی ادمینی (super user) داشته باشید و رمز عبور مناسب را وارد کنید. این ابزار فقط اکانت های کاربری را نمایش می‌دهد و اکانت های سیستمی را نه. این امکان را فراهم می‌کند تا ویژگی‌های مختلفی مانند رمز عبور کاربر را با کلیک کردن بر روی آنها تغییر دهید، اما کاربرد آن به عنوان یک ابزار مدیریت اکانت محدود است.

نکته: رمز عبورها به طور معمول در محیط گرافیکی کاربری به صورت نقاط یا ستاره‌ها نمایش داده می‌شوند به عنوان یک ویژگی امنیتی.

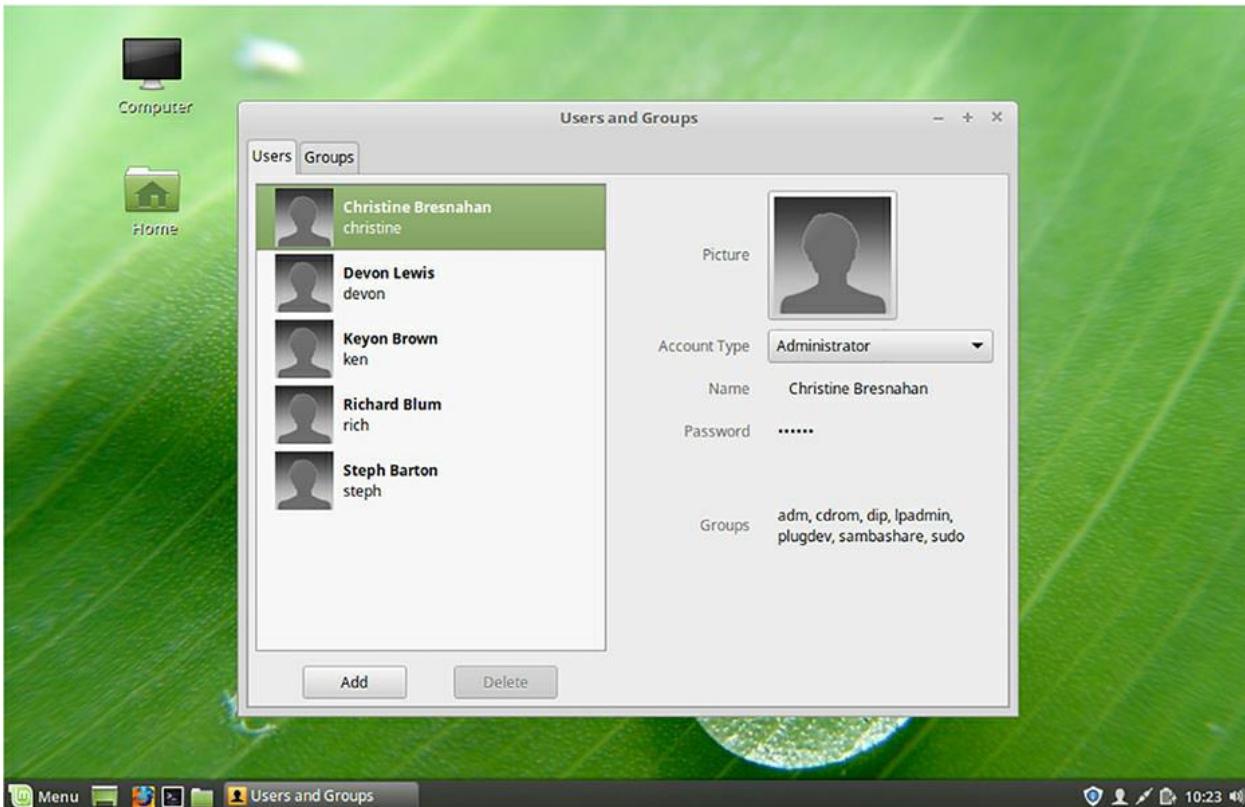


Figure 12.2 The Users and Groups account tool provides minimal account information.

شما می‌توانید تمام حساب‌های کامپیوتر را با مشاهده محتويات فایل `/etc/passwd` با استفاده از دستور `cat` یا `less` شناسایی کنید. این عمل باز همه اکانت‌ها را نمایش می‌دهد، شامل اکانت‌های سیستمی و کاربری.

در صورتی که به دنبال اطلاعات خاصی از یک اکانت باشید، می‌توانید از `grep` استفاده کنید تا آن را در `/etc/passwd` پیدا کنید، مثلاً `grep rich /etc/passwd` برای یافتن اطلاعاتی درباره هر اکانتی که به کاربری با نام کاربری rich متصل شده است. (این مثال خاص فرض می‌کند که رشته rich در فایل `passwd` وجود دارد، البته).

**نکته:** می‌توانید با استفاده از دستور `getent`، به صورت مستقیم سوابق فردی را بیرون بکشید. فقط کافی است نام کاربری را به انتهای دستور اضافه کنید، به عنوان مثال، `getent passwd rich`.

یک جایگزین که شباهت زیادی به مرور فایل `/etc/passwd` دارد، تایپ کردن دستور `getent passwd` است. دستور `getent` ورودی‌ها را از بانک‌های اطلاعاتی مدیریتی خاص، شامل فایل `/etc/passwd`، بازیابی می‌کند. در اغلب موارد، تایپ کردن `getent passwd` نتایجی مشابه تایپ کردن `cat /etc/passwd` را تولید می‌کند؛ با این حال، گاهی اوقات دو دستور مشابه نیستند. فایل `/etc/passwd` فقط اکانت‌های لوکال را تعریف می‌کند. امکان دارد که لینوکس را برای استفاده از پایگاه داده اکانت‌های شبکه تنظیم کنید تا برخی یا

همه اکانت های خود را تعریف کنید. اگر از چنین پیکربندی استفاده کنید، تایپ کردن 'getent passwd' هم اکانت های محلی و هم اکانت های تعریف شده در سرور شبکه را بازمی‌گرداند.

## Real World Scenario

### Network Account Databases

بسیاری از شبکه‌ها از پایگاه‌های داده حساب شبکه استفاده می‌کنند. این سیستم‌ها ممکن است شامل پروتکل دسترسی به دایرکتوری سبک (LDAP)، حوزه‌های Kerberos و دامنه Active Directory باشند. همه این سیستم‌ها مدیریت پایگاه‌های داده حساب را به یک کامپیوتر مرکزی (معمولاً با یک یا چند سیستم پشتیبان) یا پخش شده در سراسر یک سیستم مشخص منتقل می‌کنند. مزیت این روش این است که کاربران و مدیران نیازی به مدیریت مستقل اکانت‌ها روی چندین کامپیوتر ندارند. یک پایگاه داده اکانت می‌تواند اکانت‌ها را بر روی دهها (حتی صدها یا هزاران) کامپیوتر مختلف مدیریت کند، که وظایف مدیریتی روزانه را به طور چشمگیری ساده‌تر می‌کند و همچنین زندگی کاربران را ساده‌تر می‌کند. با استفاده از چنین سیستمی، اکثر اکانت‌های کاربری در فایل‌های `/etc/passwd` و `/etc/shadow` قرار نخواهد گرفت و گروه‌ها ممکن است در فایل `/etc/group` نیز وجود نداشته باشند (که بعداً در "Groups Understanding" توضیح داده می‌شود). با این حال، این فایل‌ها همچنان اطلاعاتی راجع به حساب‌ها و گروه‌های سیستم محلی را نگه می‌دارند.

لینوکس قادر به شرکت در این سیستم‌ها است. در واقع، برخی توزیع‌ها گزینه‌هایی را فراهم می‌کنند تا پشتیبانی از این نوع سیستم را در زمان نصب سیستم عامل فعال کنند. به طور معمول، شما باید نام یا آدرس IP سروری که پایگاه داده اکانت شبکه را میزبانی می‌کند را بدانید و باید بدانید که این سیستم از چه پروتکلی استفاده می‌کند. ممکن است نیاز به رمز عبور یا دیگر اطلاعات خاص پروتکل داشته باشید و سرور باید تنظیم شود تا دسترسی‌های از سیستم لینوکسی که شما در حال پیکربندی آن هستید، قبول کند.

فعال‌سازی استفاده از این پایگاه داده حساب شبکه پس از نصب لینوکس یک موضوع پیچیده است که در این کتاب پوشش داده نشده است. این سیستم‌ها اغلب رفتار ابزارهای مانند `passwd` و `usermod` در فصل 13 شرح داده شده است) را به صورت‌های چشمگیر یا غیر چشمگیر تغییر می‌دهند. اگر نیاز به استفاده از چنین سیستمی دارید، باید مستندات خاص سرویسی را که قصد استفاده از آن را دارید، مشاهده کنید.

### Understanding Groups

به عنوان ذکر شده قبل، گروه‌ها مجموعه‌هایی از اکانت‌هایی هستند که در فایل `/etc/group` تعریف می‌شوند. مانند `/etc/group`، فایل `/etc/passwd` شامل خطوط (رکوردها) با جداگانه دونقطه است، هر کدام یک گروه را تعریف می‌کنند. یک مثال مانند این است:

Users:x:100:games, christine

فیلدهای موجود در فایل /etc/group به شرح زیر است:

**Group Name**: فیلد اول، که در مثال قبلی users است، نام گروه است. شما از این نام در اکثر دستوراتی که به داده‌های گروه دسترسی دارند یا آنها را تغییر می‌دهند، استفاده می‌کنید.

**Password**: گروه‌ها، مانند کاربران، می‌توانند دارای گذرواژه باشند. مقدار x به این معنی است که گذرواژه در جای دیگری تعریف شده است (اما ممکن است غیرفعال شده باشد)، و یک فیلد گذرواژه خالی نشان می‌دهد که گروه دارای گذرواژه‌ای نیست.

نکته: استفاده و مدیریت گذرواژه‌های گروه یک موضوع است که خارج از دامنه این کتاب است. با این حال، اغلب حرفلهای امنیت از استفاده از گذرواژه‌های گروه منع دارند.

**GID**: لینوکس برای استفاده داخلی از مقادیر GID، مانند مقادیر UID، استفاده می‌کند. ترجمه به و از نامهای گروه برای راحتی کار کاربران و مدیران توسط سیستم انجام می‌شود.

**لیست کاربر**: می‌توانید کاربرانی را که به گروه تعلق دارند را در یک لیست جدا شده با کاما در انتهای رکورد /etc/group مشخص کنید.

مهم است که تشخیص داد که کاربران می‌توانند به دو روش عضو یک گروه تشخیص داده شوند:

1. با مشخص کردن GID گروه در ورودی‌های فردی کاربران در /etc/passwd: زیرا /etc/passwd فقط جا برای یک مقدار GID دارد، فقط یک گروه می‌تواند به این شکل تعریف شود که گروه اصلی (یا پیش‌فرض) کاربر است.

2. با مشخص کردن نامهای کاربری در لیست کاربر در فایل /etc/group: یک کاربر می‌تواند در چندین مورد در /etc/group ظاهر شود و یک گروه می‌تواند با انتساب چندین کاربر به این شیوه با آن مرتبط باشد. اگر یک کاربر به این روش با گروه مرتبط باشد اما از طریق ورودی /etc/passwd کاربر مشخص نشود، این انجمن ثانویه است.

زمانی که شما فایل‌های جدیدی ایجاد می‌کنید، این فایل‌ها به گروه کنونی شما مرتبط می‌شوند. هنگام ورود به سیستم، گروه کنونی شما به گروه اصلی شما تنظیم می‌شود. اگر می‌خواهید فایل‌هایی را ایجاد کنید که با یک گروه دیگر که به آن تعلق دارید مرتبط باشند، می‌توانید از دستور newgrp استفاده کنید، به شرح زیر:

```
$ newgrp project1
```

این دستور باعث می‌شود که گروه 1project به عنوان گروه کنونی شما تنظیم شود، بنابراین فایل‌هایی که شما ایجاد می‌کنید، با آن گروه مرتبط باشند. مالکیت گروهی فایل‌ها در امنیت فایل بسیار مهم است که جزئیات بیشتری در بخش 14 کتاب توضیح داده شده است.

## Using Account Tools

چندین دستور می‌تواند به شما کمک کند تا در مورد کاربران و گروه‌های موجود در کامپیوتر خود اطلاعات بیابید. بهویژه ابزارهای whoami و id می‌توانند به شما اطلاعاتی درباره هویت خودتان بدهند و ابزارهای who و w می‌توانند اطلاعاتی درباره اینکه چه کسانی در حال حاضر از کامپیوتر استفاده می‌کنند را نمایش دهند.

## Discovering Your Own Identity

اگر شما برای خود چندین اکانت دارید و یادتان نمی‌آید کدام یک را برای ورود به سیستم استفاده کردید، ممکن است در مورد وضعیت کنونی خود گیج شوید. در چنین موقعی، دستور whoami می‌تواند به شما کمک کند. این دستور شناسه کاربری فعلی شما را نمایش می‌دهد:

```
$ whoami
```

```
christine
```

این مثال نشان می‌دهد که اکانت فعلی christine است. اگر به اطلاعات بیشتری نیاز دارید، می‌توانید از ابزار id استفاده کنید:

```
$ id
```

```
uid=1002(christine) gid=100(users) groups=100(users)[...]
```

این مثال اطلاعاتی راجع به کاربران و گروه‌ها نشان می‌دهد:

- شناسه کاربری شما و نام کاربری: (christine) uid=1002 در این مثال
- گروه فعلی شما: (users) gid=100 در این مثال
- تمام اعضای گروه‌های شما: ورودی‌هایی که پس از groups= در این مثال آمده است

دستور id همچنین مقادیر عددی UID و GID و در صورت وجود نام‌های مرتبط را نمایش می‌دهد. گروه فعلی، گروهی است که فعال است، یا به عنوان گروه پیش‌فرض یا به دلیل استفاده از دستور newgrp.

نکته: در برخی از توزیع‌ها، دستور id اطلاعات بیشتری نسبت به مثال‌های ما نمایش می‌دهد.

شما می‌توانید خروجی دستور id را با مشخص کردن گزینه‌های مختلف محدود کنید، همانطور که در جدول 12.1 خلاصه شده است. علاوه بر این، می‌توانید نام کاربری را مشخص کرده و مانند `id rich` اطلاعات این کاربر را دریافت کنید به جای خودتان.

Table 12.1 Options for id

Long option	Short option	Effect
--group	-g	فقط ID گروه را نشان می‌دهد.
--groups	-G	تمام گروههایی که به آنها تعلق دارید را نشان می‌دهد.
--user	-U	فقط داده‌های کاربر را نمایش می‌دهد.
--name	-n	استفاده مشترک با -G، -g، یا -U؛ فقط نام را نمایش می‌دهد، بدون نمایش .GID یا .UID
--real	-r	استفاده مشترک با -G، -g، یا -U؛ فقط UID یا GID را نمایش می‌دهد، بدون نمایش نام.

## Learning Who's Online

لینوکس اجازه می‌دهد که چندین کاربر به طور همزمان به کامپیوتر دسترسی داشته باشند. اغلب این کار از طریق سرورهای دسترسی از راه دور مانند SSH انجام می‌شود، با این حال می‌توانید از ویژگی ترمینال مجازی (VT) لینوکس استفاده کنید تا با یک کیبورد و مانیتور، چندین بار وارد سیستم شویید. گاهی اوقات ممکن است بخواهید بدانید که چه کسی از کامپیوتر استفاده می‌کند، به عنوان مثال قبل از خاموش کردن کامپیوتر برای اطمینان از عدم اذیت کاربر دیگری این کار را انجام دهید.

برای اینکه بفهمید چه کسی آنلاین است می‌توانید از دستور who استفاده کنید:

```
$ who
christine    tty7    2019-08-06 10:14    (:0)
steph        tty2    2019-08-06 10:57
rich p       ts/0    2019-08-06 10:56    (192.168.0.102)
devon        tty3    2019-08-06 10:57
ken          tty4    2019-08-06 10:57
```

این مثال پنج ورودی ken و christine، steph، rich، devon را نشان می‌دهد. اطلاعات ارائه شده در خروجی پیش‌فرض شامل موارد زیر است:

Username: ستون اول خروجی دستور who نام کاربری را نمایش می‌دهد.

**Terminal Identifier**: ستون دوم خروجی دستور who کدی را نمایش می‌دهد که با ترمینال مرتبط است. در این مثال، ورود اول christine نشان می‌دهد 7tty و این ورودی ورودی GUI محلی است، اما برخی توزیع‌ها از 0 به عنوان شناسه GUI استفاده می‌کنند. ورودی‌های باقیمانده همه شناسه‌های ترمینالی از نوع pts/# یا #tty دارند که نشان دهنده جلسات متنی است. یک جلسه متنی می‌تواند یک ترمینالی باشد که در یک GUI شروع شده است، یک ورود به کنسول حالت متنی یا یک ورود از راه دور از طریق SSH یا برخی دیگر از پروتکل‌ها.

**Login Date and Time**: دستور who تاریخ و زمان ورود هر جلسه را نمایش می‌دهد. می‌توانید ببینید که جلسه christine چند دقیقه قبل از ورود steph شروع شده است.

**Remote Host**: ستون آخر خروجی دستور who، اگر وجود داشته باشد، منبع ورود را نشان می‌دهد. ورودهای کنسول (شامل ورودهای حالت متنی و مبتنی بر GUI) شامل یک منبع نمی‌شوند. یک منبع به شکل # یا # :# یا #:#، اغلب نشان دهنده یک ترمینال باز شده در یک GUI است، مانند منبع (:)christine یا (0). یک نام میزبان یا آدرس IP، مانند جلسه rich، نشان‌دهنده دسترسی از راه دور از کامپیوتر مشخص شده است.

نکته: به طور پیش‌فرض، دستور who اطلاعات خود را از فایل /var/run/utmp استخراج می‌کند.

می‌توانید با استفاده از گزینه‌های دستور who، اطلاعات اضافی را که اغلب غیر واضح یا ویژه هستند، دریافت کنید. یکی از گزینه‌هایی که احتمالاً مفیدتر از سایرین است، --count (یا -q) است که یک خلاصه فشرده‌تر از داده‌ها تولید می‌کند:

```
$ who -q
christine steph rich devon ken
# users=5
$
```

این خروجی شامل فقط نام کاربری‌ها و یک خط که تعداد کل جلسات را نشان می‌دهد. توجه کنید که تعداد کاربران تکراری با ورود چند بار به عنوان یک کاربر شمرده شده است.

نکته: توزیع‌های مختلف جزئیات کوچک متفاوتی برای دستور who دارند. با این حال، عناصر اصلی همیشه یکسان هستند.

مشابه دستور whoami اما با نمایش اطلاعات بیشتر، استفاده از آرگومان i am به دستور who اطلاعات فقط برای شناسه کاربری فعلی شما را نمایش می‌دهد.

```
$ who am i
rich pts/0 2019-08-06 10:56 (192.168.0.102)
$
```

نکته: دستور who در برخی توزیع‌ها آرگومان‌های `i` را نادیده می‌گیرد و هیچ نتیجه‌ای برنمی‌گرداند. اگر سیستم شما این آرگومان‌ها را قبول می‌کند، برای کمی طنز، می‌توانید جای آرگومان‌های `i` از آرگومان‌های `mom likes` استفاده کنید.

در این مثال، حساب کاربری فعلی rich است که به جلسه ترمینال 0/pts وارد شده است. برای کسب اطلاعات بیشتر در مورد گزینه‌ها و آرگومان‌های اضافی دستور who، به صفحه راهنمایی (man page) آن مراجعه کنید.

یک جایگزین برای دستور who دستور w است که مانند who عمل می‌کند اما خروجی کمی بیشتری ارائه می‌دهد:

```
$ w
11:17:26      up      1:10,      5 users,      load average: 0.00, 0.03, 0.04
USER          TTY          FROM          LOGIN@          IDLE JCPU PCPU WHAT
christin      tty7          :0          10:14
1:09m 17.12s 0.64s

Cinnamon    [ . . . ]

steph        tty2          10:57
4:30 0.21s 0.16s -bash

rich         pts/0          192.168.0.102 10:56      1.00s 0.20s
0.00s w

devon        tty3          10:57
20:05 0.21s 0.15s -bash

ken          tty4 10:57          19:58          0.23s
0.16s -bash

$
```

همانطور که می‌بینید، دستور w اطلاعات مشابهی به دستور who را نمایش می‌دهد، از جمله شناسه ترمینال (TTY) و زمان ورود به سیستم (به فرم دیگر). علاوه بر این، دستور w اطلاعات بیشتری را نمایش می‌دهد:

- زمان idle جلسه (Session) نشان می‌دهد که کاربر با این جلسه تعامل داشته باشد. این اطلاعات می‌تواند به شما کمک کند تا جلساتی که کاربر ممکن است ترک کرده باشد را شناسایی کنید.
- ستون CPU مجموع زمان پردازش CPU مرتبط با جلسه را نشان می‌دهد. این اطلاعات می‌تواند اطلاعات مفیدی برای اشکال زدایی باشد اگر که کامپیوتر به دلیل فرآیندهای بی‌کنترل دچار کند شده باشد.
- ستون PCPU مجموع زمان پردازش CPU مرتبط با فرآیند جاری در جلسه را نشان می‌دهد. باز هم، این اطلاعات می‌تواند به شما کمک کند تا فرآیندهای بی‌کنترل را پیدا کنید.
- ستون WHAT به شما می‌گوید که جلسه در حال اجرای چه برنامه‌ای است.
- برخی پیکربندی‌ها همچنین ستون FROM را نمایش می‌دهند که نام میزبان را نشان می‌دهد.
- استفاده از گزینه `f`-این ویژگی را روشن یا خاموش می‌کند. چندین گزینه دیگر نیز می‌تواند خروجی دستور `W` را حذف یا تغییر دهد. برای جزئیات بیشتر به صفحه راهنمایی (man page) این دستور مراجعه کنید.

## Working as root

لینوکس بر اساس یونیکس مدل‌سازی شده است که به عنوان یک سیستم عامل چند کاربره طراحی شده بود. به طور اصولی، شما می‌توانید هزاران اکانت در یک کامپیوتر یونیکس (یا لینوکس) داشته باشید. با این حال، حداقل یک کاربر نیاز به قدرت فوق العاده‌ای دارد تا ویژگی‌های کامپیوتر را به طور کلی مدیریت کند. به طور تاریخی، این اکانت root است که به عنوان کاربر فوق العاده یا مدیر نیز شناخته می‌شود. دانستن اینکه چرا root وجود دارد، چگونه کارها را به عنوان root انجام دهید (اگر مجبور باشید) و چگونه از دسترسی‌های root به صورت ایمن استفاده کنید، برای مدیریت یک سیستم لینوکس مهم است.

## Understanding User Types

بیشتر افراد از کامپیوترها برای انجام کارهای روزمره عادی استفاده می‌کنند – مرور وب، نوشتن نامه، مدیریت مجموعه موسیقی و غیره. این فعالیت‌ها به طور جمعی به عنوان user tasks شناخته می‌شوند و نیازی به دسترسی‌های ویژه ندارند. همانطور که قبل اشاره شد، یک کامپیوتر لینوکس می‌تواند اکانت‌های زیادی داشته باشد و کاربران می‌توانند از این اکانت‌ها (که به عنوان حساب‌های غیرممتاز، کاربران غیرممتاز یا کاربران استاندارد نیز شناخته می‌شوند) برای انجام این وظایف کاربری استفاده کنند.

اکانت root برای انجام وظایف مدیریتی وجود دارد. این وظایف شامل نصب نرم‌افزار جدید، آماده‌سازی یک دیسک جدید برای استفاده در کامپیوتر و مدیریت اکانت‌ها عادی است. این وظایف نیاز به دسترسی به فایل‌های سیستم دارند که کاربران عادی نیازی به تغییر یا حتی خواندن آنها ندارند.

نکته: نوع دیگری از کاربر، کاربر سیستم است. این‌ها حساب‌های غیرورود برای دیمن‌ها، سرویس‌ها یا برنامه‌ها هستند. حساب‌های کاربری سیستم معمولاً دارای شماره ID پایین، بدون رمز عبور (بنابراین حساب قفل است) و /bin/false یا /usr/sbin/nologin /sbin/nologin به عنوان شل پیش‌فرض خود هستند.

برای تسهیل انجام این وظایف، روت می‌تواند هر فایلی روی کامپیوتر را بخواند و بنویسد. از آنجا که لینوکس برای ذخیره تنظیمات سیستم به فایل‌ها متکی است، این عملًا به روت قدرت تغییر هر جزئی از عملکرد سیستم عامل را می‌دهد، که هدف داشتن یک حساب کاربری سوپر یوزر است. اگر کامپیوتر یک ایستگاه کاری باشد که فقط توسط یک فرد استفاده می‌شود، ممکن است تعجب کنید که چرا تمایز بین روت و حساب کاربری کاربر ضروری است. توضیح این است که قدرت حساب کاربری روت می‌تواند منجر به خسارت‌های تصادفی شود. به عنوان مثال، فرمان rm را در نظر بگیرید. اگر به عنوان یک کاربر عادی یک دستور rm را اشتباه تایپ کنید، ممکن است به طور تصادفی فایل‌های خودتان را حذف کنید ولی نمی‌توانید فایل‌های سیستم را حذف کنید. اما اگر همین اشتباه را به عنوان روت مرتكب شوید، ممکن است فایل‌های سیستم را حذف کنید و کامپیوتر را غیرقابل بوت کنید. بنابراین، باید هنگام استفاده از حساب کاربری روت احتیاط کنید (یا اصلًا از آن استفاده نکنید)، موضوعی که در بخش «Using root Privileges Safely» بیشتر به آن پرداخته خواهد شد.

## Acquiring root Privileges

وقتی نیاز به انجام یک وظیفه کامند لاینی دارید که نیاز به امتیازات روت دارد، می‌توانید این کار را به سه روش انجام دهید:

**Log In as root**: شما می‌توانید مستقیماً به عنوان روت در یک شل حالت متنی یا با استفاده از ابزاری مانند SSH وارد شوید. حتی می‌توانید در برخی توزیع‌های لینوکس به حالت GUI به عنوان روت وارد شوید. برخی توزیع‌ها به‌طور پیش‌فرض اجازه ورود مستقیم روت را نمی‌دهند زیرا این کار خطرناک است.

**Use su**: فرمان su به شما امکان می‌دهد که هویت خود را درون یک شل تغییر دهید. با تایپ su می‌توانید هویت خود را به هویت کاربر مشخص شده تغییر دهید. اگر username را حذف کنید، روت فرض می‌شود، بنابراین با تایپ su می‌توانید عملًا روت شوید.

با این حال، برای اینکه این فرمان کار کند باید رمز عبور اکانت هدف (روت یا هر چیز دیگری) را بدانید. بعد از اینکه به این شکل امتیازات روت را به دست آورده‌اید، می‌توانید به عنوان روت هر تعداد فرمان که بخواهید تایپ کنید. وقتی کارتان تمام شد، تایپ exit کنید تا وضعیت سوپر یوزر خود را لغو کنید.

همچنین می‌توانید از su برای اجرای یک فرمان به عنوان روت استفاده کنید. از گزینه -c استفاده کنید، به عنوان مثال su -c command را به عنوان روت اجرا کنید.

اگر درون فرمان از خط تیره (-) استفاده کنید، به عنوان مثال -su یا luke -su، برنامه یک جلسه ورود باز می‌کند که اسکریپت‌های ورود کاربر هدف را اجرا می‌کند. این می‌تواند مهم باشد زیرا این اسکریپت‌ها اغلب متغیرهای محیطی مانند \$PATH را تنظیم می‌کنند که می‌تواند برای آن کاربر مهم باشد.

نکته: su مخفف switch user or substitute user می‌باشد.

**Use sudo**: فرمان sudo مشابه su است، اما فقط برای یک فرمان در یک زمان کار می‌کند که شما آن را بعد از sudo تایپ می‌کنید، مشابه استفاده از -c su. به عنوان مثال، تایپ sudo cat /etc/shadow به شما اجازه

می‌دهد که محتوای فایل `/etc/shadow` را ببینید، که برای کاربران عادی قابل خواندن نیست. شما باید یا رمز عبور خودتان یا رمز عبور روت را تایپ کنید، بسته به تنظیمات `sudo`، هنگامی که از این برنامه استفاده می‌کنید. (هنگام استفاده از `-C`، شما همیشه باید رمز عبور روت را تایپ کنید). فرمان بعدی که تایپ می‌کنید با امتیازات حساب عادی شما اجرا خواهد شد. برخی توزیع‌ها، مانند اوبونتو و فدورا، به شدت به `sudo` متکی هستند و به طور پیش‌فرض اجازه ورود مستقیم روت را نمی‌دهند.

## Real World Scenario

### The Legalities of Acquiring root Privileges

اگر سیستم لینوکس شما در محل کار استفاده می‌شود، مهم است که سیاست‌های شرکت خود را در مورد کسب امتیازات روت مشخص کنید. ورود مستقیم به حساب روت یا استفاده از فرمان `SU` برای به دست آوردن امتیازات روت، محیط را به وجود می‌آورد که به آن محیط تکذیب‌پذیر (repudiation environment) می‌گویند. در این محیط، فرد می‌تواند اقدامات خود را انکار کند که به عنوان روت وارد شده است. این محیط می‌تواند به یک کاربر حساب روت اجازه دهد که فعالیت‌های غیرقانونی یا مشکل‌ساز انجام دهد و سپس به طور قانونی مسئولیت آن فعالیت‌ها را انکار کند. این یک وضعیت بالقوه خطرناک است.

بسیاری از شرکت‌ها سیاستی دارند (یا باید داشته باشند) که اصرار دارند فرمان `sudo` برای هر کسی که نیاز به کسب امتیازات روت دارد، استفاده شود. فرمان `sudo` فعالیت‌های کاربر را ردیابی و ثبت می‌کند و بنابراین محیط را ایجاد می‌کند که به آن محیط غیرقابل‌انکار (nonrepudiation environment) می‌گویند که در آن اقدامات نمی‌توانند به طور قانونی انکار شوند.

بسیاری از توزیع‌های مدرن لینوکس اجازه ورود مستقیم به حساب روت را نمی‌دهند. با این حال، اگر بتوانید امتیازات روت را با ورود مستقیم به عنوان روت یا با استفاده از `SU` - کسب کنید، پرامپت شل شما تغییر خواهد کرد:

```
[rey@jakku ~]$ su -
```

Password:

```
[root@jakku ~]#
```

استفاده از `SU` - به جای فقط دستور `SU` همیشه یک ایده خوب است. خط تیره (-) محیط اکانت را به درستی تنظیم می‌کند. اگر از خط تیره استفاده نکنید، ممکن است با مشکلات دستور مواجه شوید.

در این مثال، نام کاربری از `rey` به `root` تغییر کرده است و آخرین نویسه از علامت دلار (\$) به علامت هش (#) تغییر کرده است. به دلیل اینکه برای بیشتر مثال‌های چاپ شده در خطوط جداگانه در این کتاب، تنها

آخرین کاراکتر پرامپت استفاده شده است، اینگونه مثال‌ها به طور ضمنی نشان می‌دهند که آیا یک دستور به احتیاج دارد که از امتیازات root استفاده کند. به عنوان مثال، در نظر بگیرید دسترسی به فایل /etc/shadow که قبلاً ذکر شده است:

```
# cat /etc/shadow
```

استفاده از علامت هش در پرامپت نشان می‌دهد که شما باید این دستور را با امتیازات root تایپ کنید.

**نکته:** برای استفاده از دستورات root در Linux Mint و Fedora Workstation یا مانند sudo cat /etc/shadow یا sudo su (sudo cat /etc/shadow) را تایپ کنید تا یک شل root دارای مدت زمان طولانی‌تر را فراهم کنید.

بعضی از فصل‌های این کتاب هر دو روش مدیریت سیستم را به صورت گرافیکی و حالت متنی شرح داده‌اند. پس چگونه می‌توانید لینوکس را در حالت GUI مدیریت کنید اگر باید از دستور sudo در حالت متنی برای به دست آوردن امتیازهای root استفاده کنید؟ بیشتر توزیع‌ها به شما امکان می‌دهند تا ابزارهای مدیریتی را از منوهای دسکتاپ کامپیوترها اجرا کنید و این ابزارهای GUI هنگام نیاز به امتیازهای مدیریتی از شما رمز عبور سوپر کاربر را می‌پرسند، همان‌طور که در شکل 12.3 نشان داده شده است. اگر رمز عبور را به درستی وارد کنید، برنامه ادامه خواهد داد. نتیجه مشابه اجرای برنامه از یک شل با استفاده از sudo است.

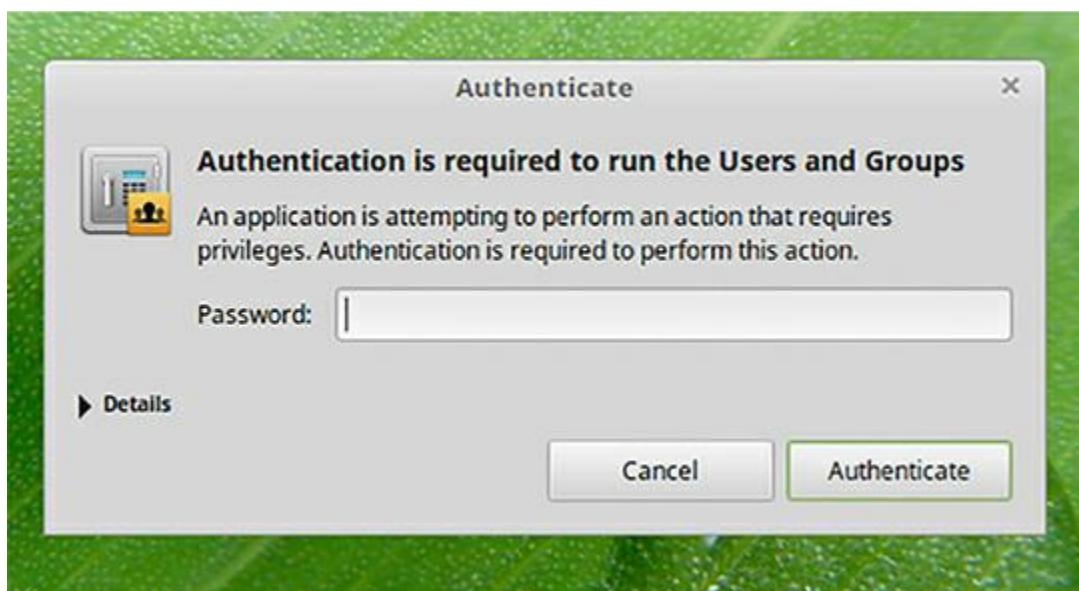


Figure 12.3 Administrative tools in the GUI ask for a password when administrator privileges are needed.

## Using root Privileges Safely

همان‌طور که قبلاً گفته شد، دسترسی به قدرت روت خطرناک است. ممکن است به طور اتفاقی فایل‌های برنامه‌های حیاتی را پاک کنید و باعث ساعتها زمان برای بازگشت سیستم شوید. تصور کنید اگر به اشتباه یک فایل پیکربندی مهم را خراب کنید یا یک مجموعه از پشتیبان‌های مهم را از بین ببرید. همه اشتباه می‌کنند - برخی از این اشتباهات می‌توانند برای یک شرکت کاملاً ویرانگر باشند. تصور کنید هکرها دسترسی روت به کامپیوتر شما را پیدا کرده‌اند: تغییرات ناخواسته در فایل‌های پیکربندی، آسیب به برخی از فایل‌های سیستم (حتی اگر همه نه)، و تغییرات در مالکیت یا مجوزهای دسترسی بر روی فایل‌های کاربر عادی که باعث عدم دسترسی به صاحبان واقعی آن‌ها می‌شود. هنگام نیاز به دسترسی روت، باید اقدامات احتیاطی زیر را انجام دهید:

- پرسیدن از خود آیا واقعاً به دسترسی روت نیاز دارید. گاهی اوقات راهی برای دستیابی به هدف بدون دسترسی به حقوق ادمین یا با استفاده از این حقوق به یک روش محدودتر از آنچه که قرار بوده‌اید وجود دارد. به عنوان مثال، ممکن است متوجه شوید که فقط روت می‌تواند به یک دیسک جابجایی دسترسی داشته باشد. چنین مشکلی معمولاً با تنظیم مجوزها بر روی دیسک به یکی از روش‌های مختلف حل می‌شود که از روت محدود می‌شود.
- قبل از فشردن کلید Enter پس از تایپ هر دستور به عنوان روت (یا کلیک بر روی دکمه تأیید در یک برنامه GUI که به عنوان روت اجرا می‌شود)، دست خود را از کیبورد و موس بردارید، دستور را مرور کنید و مطمئن شوید که از هر نظر صحیح است. یک اشتباه ساده می‌تواند عواقب بسیاری داشته باشد.
- هرگز برنامه مشکوکی را به عنوان روت اجرا نکنید. در سیستم‌های چندکاربره، کاربران ناسالم ممکن است سعی کنند مدیران را فریب دهند تا برنامه‌هایی را اجرا کنند که کارهای ناپسندی انجام می‌دهند یا اختیارات روت را به حمله‌کننده می‌دهند. برنامه‌هایی که از سایت‌های اینترنتی تصادفی دانلود می‌شوند در اصل ممکن است برای آسیب دادن به امنیت شما طراحی شوند و این برنامه‌ها وقتی به عنوان روت اجرا می‌شوند، بسیار خطرناک‌تر هستند.

نکته: اگر یک برنامه از شما رمز عبور خود یا رمز عبور روت را درخواست می‌کند و این برنامه یک برنامه مدیریتی معتبر نیست که به آن اعتماد دارید، باید مشکوک باشید! قبل از وارد کردن رمز عبور، برنامه را بررسی کنید تا اطمینان حاصل کنید که معتبر و ایمن است.

- استفاده از امتیازات روت را به مدت کوتاهی که امکان پذیر است. اگر فقط نیاز به تایپ یک یا دو دستور به عنوان روت دارید، این کار را انجام دهید و سپس دستور exit را در شل روت تایپ کنید تا از حساب کاربری روت خارج شوید و یا به امتیازات عادی خود بازگردید. بهتر است از دستور sudo برای اجرای دستورات استفاده کنید. آسان است که فراموش کنید که از شل روت استفاده می‌کنید و بنابراین دستورات را به عنوان روت تایپ کنید که نیاز به این امتیاز را ندارند.
- هر دستوری که به عنوان روت تایپ می‌شود، یک ریسک است. هرگز شل روت را برای دیگران قابل دسترسی نگذارید. اگر وظایف نگهداری روت را انجام می‌دهید و تماسی به شما زده شود، قبل از ترک کامپیوتر دستور exit را در شل روت تایپ کنید.

• با رمز عبور روت محتاط باشید. رمز عبور را با دیگران به اشتراک نگذارید و در محیط عمومی یا زمانی که دیگران ممکن است نگاه کنند، دقت کنید. اگر از لینوکس به صورت حرفه‌ای استفاده می‌کنید، کارفرمای شما ممکن است دستورالعمل‌هایی درباره کسانی که می‌توانند به امتیاز روت در کامپیوتر دسترسی داشته باشند، داشته باشد. این قوانین را بیاموزید و آنها را رعایت کنید! اطمینان حاصل کنید که یک رمز عبور قوی برای روت انتخاب کنید.

نکته: فصل 13 در رابطه با چگونگی انتخاب پسورد توضیح میدهد.

در پیروی از این قوانین کلی، می‌توانید از آسیب دیدن کامپیوتر خود یا دادن دسترسی روت به شخص دیگری جلوگیری کنید.

# CHAPTER 13

## Creating Users and Groups

لینوکس یک سیستم عامل چندکاربره است، به این معنی که یک کامپیوتر لینوکس می‌تواند از چندین کاربر پشتیبانی کند، هر یک با یک حساب کاربری منحصر به فرد. با این قابلیت، نیاز به مدیریت حساب‌های کاربران به وجود می‌آید و این فصل روش‌هایی را که برای انجام این کار استفاده خواهید کرد، پوشش می‌دهد.

این فصل با اطلاعاتی در مورد نحوه ایجاد حساب‌ها آغاز می‌شود. با ایجاد حساب‌ها، سپس باید بدانید که چگونه این حساب‌ها را اصلاح کنید و در صورت لزوم، آن‌ها را حذف کنید. در نهایت، گروه‌ها از بسیاری جهات مشابه حساب‌ها هستند، بنابراین یاد خواهید گرفت که چگونه گروه‌ها را ایجاد و مدیریت کنید.

### Creating New Accounts

در بسیاری از محیط‌ها، وظیفه افزودن حساب‌ها بسیار رایج است. کسب‌وکارهای بزرگ کارمندان جدید استخدام می‌کنند، دانشگاه‌ها دانشجویان جدید می‌پذیرند، سازمان‌های خیریه داوطلبان جدید جذب می‌کنند و غیره. بنابراین، باید بدانید که چگونه حساب‌های جدید ایجاد کنید. اما ابتدا، این بخش به مسائل مهمی مانند تصمیم‌گیری در مورد چگونگی استفاده از گروه‌ها و انتخاب یک رمز عبور خوب می‌پردازد. سپس یاد خواهید گرفت که چگونه با استفاده از ابزارهای گرافیکی و حالت متنی حساب‌ها را ایجاد کنید.

### Deciding on a Group Strategy

همانطور که در فصل 12، "Understanding Basic Security" توضیح داده شد، گروه‌های لینوکس مجموعه‌هایی از کاربران هستند. شما می‌توانید از گروه‌ها برای کنترل دسترسی افراد به فایل‌های خاص استفاده کنید. همانطور که در فصل 14، "Setting Ownership and Permissions" توضیح داده خواهد شد، افراد می‌توانند وابستگی‌های گروهی و مجوزهای گروهی فایل‌های خود را تغییر دهند. بنابراین، نحوه استفاده شما از گروه‌ها می‌تواند استراتژی کلی امنیت کامپیوتر شما را تحت تاثیر قرار دهد. دو رویکرد رایج وجود دارد:

**User Groups:** هر کاربر می‌تواند یک گروه مرتبط داشته باشد؛ به عنوان مثال، کاربر luke می‌تواند یک گروه به نام luke داشته باشد. این کاربر سپس می‌تواند مالکیت گروهی فایل‌های خود را به luke تنظیم کند یا مجوزهای گروهی را به هر چیزی که دلخواه است تنظیم کند و مدیر سیستم می‌تواند کاربران را به گروه luke اضافه کند. از آن پس، اعضای گروه luke می‌توانند با استفاده از مجوزهایی که توسط کاربر luke تعیین شده است، به فایل‌های این گروه دسترسی پیدا کنند. این رویکرد بر کنترل دسترسی به فایل‌های فردی کاربران تأکید دارد.

**Project Groups:** در این روش، شما گروه‌هایی بر اساس پروژه‌های کاری، وابستگی‌های دپارتمانی یا دیگر گروه‌بندی‌های واقعی کاربران ایجاد می‌کنید. به عنوان مثال، ممکن است یک گروه به نام sales برای کاربران در بخش فروش داشته باشید. اعضای این گروه که می‌خواهند فایل‌ها را با دیگر اعضای این گروه به اشتراک

بگذارند، مالکیت و مجوزهای گروهی را به طور مناسب تنظیم کرده و فایل‌ها را در مکانی توافق شده ذخیره می‌کنند. این رویکرد زمانی که کامپیوتر توسط تعداد زیادی از افراد که در گروه‌های به راحتی تعریف شده همکاری می‌کنند استفاده می‌شود، بهترین عملکرد را دارد.

این دو رویکرد متقابلاً انحصاری نیستند؛ شما می‌توانید آنها را ترکیب کرده یا رویکرد خود را ایجاد کنید. همچنین باید بدانید که کاربران می‌توانند عضو چندین گروه باشند. در واقع، این برای کار کردن رویکرد گروه‌های کاربری ضروری است؛ در غیر این صورت، گروه‌ها با حساب‌ها همپوشانی دارند.

نکته: به طور پیش‌فرض، برخی توزیع‌ها از استراتژی گروه‌های کاربری و برخی دیگر از استراتژی گروه‌های پروژه‌ای استفاده می‌کنند. در حالت دوم، بیشتر کاربران به طور پیش‌فرض در گروهی به نام users یا چیزی مشابه آن قرار می‌گیرند.

اگر از روش project group استفاده می‌کنید، باید به این فکر کنید که کدام گروه باید گروه اصلی کاربر جدید باشد. این گروهی است که به طور پیش‌فرض مالکیت گروهی فایل‌های کاربر را به آن اختصاص خواهد داد.

## Selecting a Good Password

وقتی یک اکانت ایجاد می‌کنید، معمولاً باید یک رمز عبور برای آن ایجاد کنید. گاهی اوقات، کاربر می‌تواند رمز عبور را هنگام ایجاد حساب انتخاب کند. در موقع دیگر، شما باید یک رمز عبور انتخاب کنید که کاربر در ابتدا از آن استفاده کند. در چنین مواردی، به کاربر دستور دهید که هر چه سریع‌تر رمز عبور را تغییر دهد. در هر دو حالت، مهم است که کاربران را در مورد انتخاب یک رمز عبور مناسب آگاه کنید.

نکته: حتماً خودتان نیز از توصیه‌های این بخش پیروی کنید، بهویژه برای رمز عبور root اگر از حساب کاربری روت استفاده می‌کنید!

- رمزهای عبور ضعیف ولی رایج شامل موارد زیر می‌شوند:
- نام اعضای خانواده، دوستان و حیوانات خانگی
- کتاب‌ها، فیلم‌ها، برنامه‌های تلویزیونی محبوب یا شخصیت‌های آنها
- شماره‌های تلفن، آدرس‌های خیابان، یا شماره‌های شناسنامه
- هر اطلاعات شخصی معنی‌دار دیگر
- هر کلمه‌ای که در یک فرهنگ لغت یافت می‌شود (به هر زبانی)
- هر ترکیب ساده‌ای از صفحه‌کلید یا عددی، مانند qwerty یا 123456

بهترین رمزهای عبور ممکن، مجموعه‌های تصادفی از حروف، اعداد و علائم نگارشی هستند. متأسفانه، چنین رمزهایی سخت به یاد سپرده می‌شوند. یک سازش معقول این است که رمز عبور را در دو مرحله بسازید:

1. انتخاب یک پایه که به راحتی به یاد سپرده شود ولی حدس زدن آن دشوار باشد.
2. اصلاح آن پایه به گونه‌ای که سختی حدس زدن رمز عبور را افزایش دهد.

یک رویکرد برای ساختن یک پایه این است که از دو یا چند کلمه نامرتبط استفاده کنید، مانند نان و خودکار (bunpen). سپس می‌توانید این دو کلمه را ترکیب کنید (bunpen). یک رویکرد دیگر، و احتمالاً بهتر از اولی، استفاده از حروف اول یک عبارت معنادار برای کاربر است. به عنوان مثال، حروف اول عبارت "yesterday I went to the dentist" می‌شود. در هر دو حالت، پایه نباید کلمه‌ای در هیچ زبانی باشد. به طور کلی، هرچه رمز عبور طولانی‌تر باشد، بهتر است.

**نکته:** بسیاری از توزیع‌ها حداقل طول رمز عبور را تعیین می‌کنند، مانند شش یا هشت کاراکتر.

با داشتن پایه، اکنون زمان تغییر آن برای ایجاد رمز عبور است. کاربر باید حداقل دو تغییر از تغییرات ممکن را اعمال کند:

**افزودن اعداد یا علامت‌گذاری:** یکی از تغییرات مهم افزودن اعداد یا علامت‌های تصادفی به پایه است. این مرحله ممکن است به عنوان مثال،  $n?y+i9wttd$  یا  $bu3npe?$  را ایجاد کند. به طور کلی، حداقل دو نماد یا عدد اضافه کنید.

**ترکیب حروف بزرگ و کوچک:** لینوکس از رمزهای عبور حساس به حروف بزرگ و کوچک استفاده می‌کند، بنابراین ترکیب حروف بزرگ و کوچک می‌تواند امنیت را افزایش دهد. اعمال این قانون ممکن است به  $y+i9WttD$  و  $Bu3nPE?n$  منجر شود.

**برعکس کردن ترتیب:** تغییری که به تنها بسیار ضعیف است اما می‌تواند در کنار سایر تغییرات به امنیت کمک کند، معکوس کردن ترتیب برخی یا تمام حروف است. شما می‌توانید این تغییر را فقط به یک کلمه از یک پایه دو کلمه‌ای اعمال کنید. این می‌تواند به  $DttW9i+y$  و  $Bu3nn?EP$  منجر شود.

**افزایش اندازه رمز عبور:** تلاش برای کشف رمز عبور توسط یک نفوذگر به جستجوی یک سوزن در یک انبار کاه تشبيه شده است. یکی از راه‌های سخت‌تر کردن این وظیفه افزایش اندازه انبار کاه است. در اصطلاح رمز عبور، این به معنای افزایش اندازه رمز عبور است. شما می‌توانید این کار را با استفاده از کلمات یا عبارات بزرگ‌تر انجام دهید، البته این می‌تواند رمز عبور را دشوارتر به خاطر سپردن و تایپ کردن کند. حتی یک افزایش اندازه که تنها یک کاراکتر را تکرار می‌کند می‌تواند مفید باشد. بنابراین شما ممکن است رمزهای عبور را به  $y+i9WttD$  یا  $Bu3nn?EPiiiiiiiiii$  تبدیل کنید.

**نکته:** مؤسسه ملی استاندارد و فناوری (NIST) یک آزمون غیرقانون‌گذار آمریکایی است. آنها بسیاری از توصیه‌های مربوط به تجارت را منتشر می‌کنند، از جمله امنیت کامپیوترهای کسب و کار. پیشنهادات NIST معمولاً به رویه‌هایی برای شرکت‌ها و سازمان‌های دولتی تبدیل می‌شوند. در سال 2017، NIST دستورالعمل‌های جدیدی در مورد رمزهای عبور امن صادر کرد و به طرز شگفت‌انگیزی دیگر رمزهای عبور طولانی و پیچیده توصیه نمی‌شوند. با این حال، ممکن است مدت زمان زیادی طول بکشد تا دستورالعمل‌های جدید رمز عبور NIST وارد دنیای روزمره شما شوند، اگر اصلاً وارد شوند.

بهترین ابزار شما برای تشویق کاربران به انتخاب رمزهای عبور قوی، آموزش آنها است. در ادامه، برخی از مطالباتی که برای کاربران مفید است به آنها آموزش دهید را بررسی می‌کنیم:

- رمزهای عبور ممکن است توسط افراد بدخواه که آنها را میشناسند یا حتی آنها را هدف قرار دادند و اطلاعات شخصی آنها را از طریق رسانه‌های اجتماعی، دایرکتوری‌های تلفن و بازاریابی، و غیره جستجو کردند حدس زده شود.
  - اگرچه لینوکس پسوردهای خود را داخلی salted و هش می‌کند، برنامه‌هایی وجود دارند که تمام دیکشنری‌ها را از طریق الگوریتم‌های salting و هش پسورد لینوکس برای مقایسه با پسوردهای سیستم لینوکس می‌فرستند. اگر تطابق وجود داشته باشد، پسورد پیدا شده است.
  - اکانت‌ها ممکن است به عنوان گام اول برای تخریب کامپیوتر به کار گرفته شود یا به عنوان نقطه شروع حملات به سایر کامپیوترها.
  - کاربران هرگز نباید پسوردهای خود را به دیگران فاش کنند، حتی افرادی که ادعا می‌کنند مدیران سیستم هستند. این یک کلاهبرداری معمول است، زیرا مدیران واقعی سیستم به پسوردهای کاربران نیاز ندارند.
  - هرگز نباید از همان رمزعبور بر روی چندین سیستم استفاده کنید، زیرا این کار منجر به تبدیل یک نفوذ در یک کامپیوتر به نفوذ در همه آنها خواهد شد.
  - نوشتن رمزهای عبور یا ارسال آنها از طریق ایمیل، روش‌های پرخطری هستند. نوشتن رمز عبور روی یک برگه و چسباندن آن به مانیتور کامپیوتر بسیار احمقانه است.
- گفتن این اطلاعات به کاربران خود کمک می‌کند تا دلایل نگرانی شما را درک کنند، و احتمالاً باعث تحریک حداقل برخی از آنها برای انتخاب رمزهای عبور قوی می‌شود.

## Don't Use These Passwords!

اگر شما در اینترنت راجع به پسوردهای رایج سرچ کنید، به سرعت وبسایت‌هایی را پیدا خواهید کرد که نتایج نظرسنجی‌هایی از معمول‌ترین و آسان‌ترین رمزهای عبوری که پژوهشگران امنیتی کشف کرده‌اند، نمایش می‌دهند. جزئیات از یک نظرسنجی به نظرسنجی دیگر متفاوت است، اما به طور معمول رمزهای عبور متداول شامل موارد زیر می‌شود:

- 123456
- password
- 12345678
- qwerty
- 111111
- sunshine
- iloveyou
- princess
- football
- password1

اینگونه رمزهای عبور به راحتی توسط برنامههای حدس زدن رمز عبور با نیروی خشن کشف می‌شوند و در مجموعه‌هایی از رمزهای عبور که در اینترنت منتشر می‌شوند نیز وجود دارند. استفاده از چنین رمز عبوری به سختی بهتر از استفاده از هیچ رمز عبوری است. خود را محافظت کنید و یک رمز عبور بهتر بسازید!

## Creating Accounts Using GUI Tools

حال که شما تا حدودی ایده‌ای از نوع سیاست گروهی که می‌خواهید استفاده کنید و چگونگی ایجاد یک رمز عبور خوب دارید، می‌توانید شروع به ایجاد حساب‌ها کنید. برخی توزیع‌های لینوکس اجازه می‌دهند که این کار را حداقل به طور جزئی از طریق ابزار GUI انجام دهید. این ابزارها در توزیع‌های مختلف لینوکس متفاوت هستند. تفاوت‌های شامل چگونگی دسترسی به این ابزارها و نام ابزارها می‌شود. در برخی از توزیع‌ها، با استفاده از ابزار جستجوی دسکتاپ، می‌توانید user را تایپ کنید تا ابزار مناسب برای ایجاد حساب کاربری را پیدا کنید. در دیگر توزیع‌ها، شما باید از منوهای دسکتاپ برای پیدا کردن ابزار صحیح استفاده کنید. نام ابزار ممکن است مانند User Accounts، Users and Groups Administration یا User and Groups باشد.

نکته: وقتی که ابزار Users and Groups Administration را در لینوکس مینت اجرا می‌کنید، یک پنجره‌ی دیالوگ ظاهر می‌شود که شما را می‌طلبد تا رمز عبور حساب کاربری خود را وارد کنید تا بتوانید ادامه دهید. اگر دسترسی‌های superuser privileges را مشاهده خواهید کرد که در شکل 13.1 نشان داده شده است.

به عنوان نمونه‌ای از ابزار User and Groups management، در شکل 13.1 ابزار Administration لینوکس مینت نمایش داده شده است. با استفاده از این ابزار خاص، شما می‌توانید بیشتر از افزودن حساب کاربری انجام دهید. با این حال، تمرکز اصلی در اینجا بر روی ایجاد حساب‌ها است.

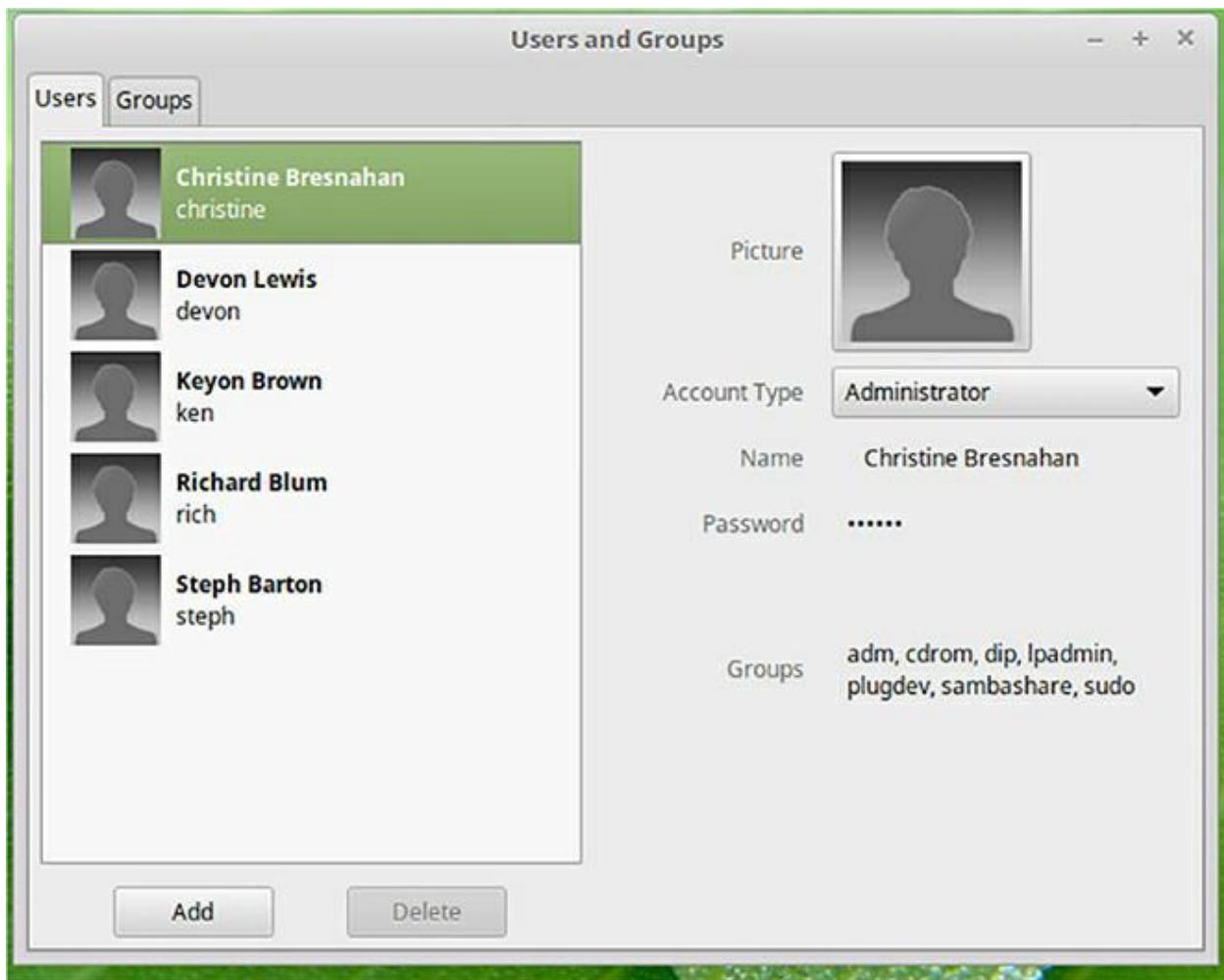


Figure 13.1 The Linux Mint User and Groups Administration utility provides many options for creating and managing accounts.

برای افزودن یک یوزر با ابزار User and Group Administration، این مراحل را دنبال کنید:

1. بر روی دکمه Add کلیک کنید. نتیجه این کار، دیالوگی است که در شکل 13.2 نمایش داده شده است.
2. نام کامل کاربر را در فیلد User's Full Name وارد کنید. این ورودی در فیلد comment فایل /etc/passwd ذخیره می‌شود و ممکن است در ابزارهای مختلف نمایش داده شود؛ به عنوان مثال، در برخی محیط‌های دسکتاپ ظاهر می‌شود هنگامی که کاربر به دسکتاپ وارد می‌شود.
3. نام کاربری را در فیلد Username وارد کنید. این نام کاربری است که کاربر در دستورات ورود به لینوکس تایپ می‌کند.
4. برای پایان دادن به ایجاد حساب، بر روی دکمه Add کلیک کنید.
5. بر روی نام کاربری کلیک کنید، سپس بر روی عبارت "no password set" در توضیحات حساب کلیک کنید تا دیالوگ تغییر رمز عبور نمایش داده شود که در شکل 13.3 نمایش داده است.

- .6. رمز عبور را دو بار وارد کنید، یکبار در فیلد Password و دوباره در فیلد Confirm Password
- .7. هنگامی که ابزار رمز عبور را به اندازه کافی قوی می‌داند، دکمه Change (که خاکستری نیست) برای شما در دسترس است تا برای اعمال تغییرات رمز عبور کلیک کنید.

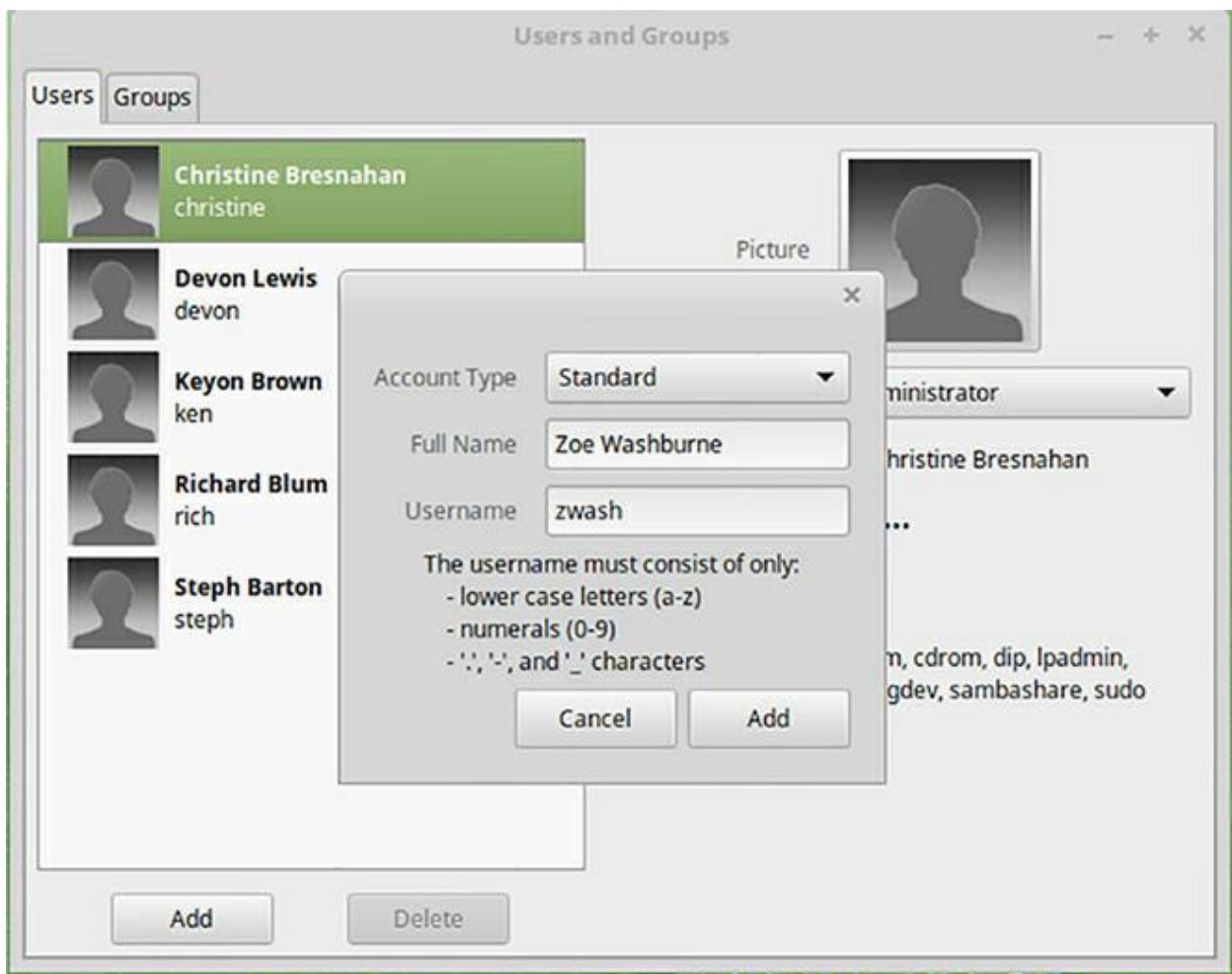


Figure 13.2 You can enter all the basic account information using this dialog.

حساب کاربری جدید در لیست تب کاربران ظاهر می‌شود. می‌توانید بعداً آن را تغییر یا حذف کنید، همانطور که در ادامه این فصل توضیح داده شده است.

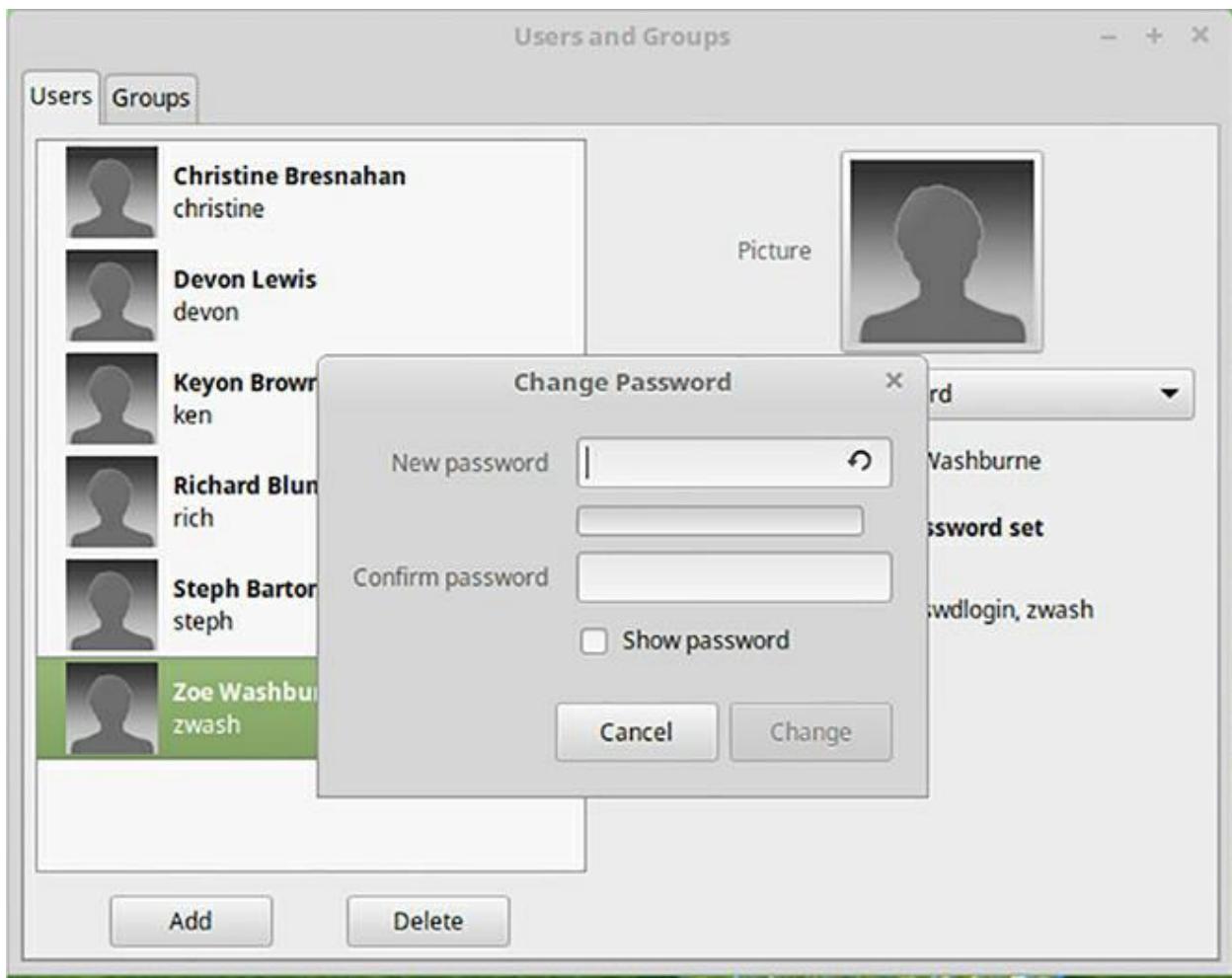


Figure 13.3 Set the user account's password using the Change Password dialog.

## Creating Accounts from the Shell

با هر توزیعی، شما از ابزار useradd برای ایجاد یک حساب کاربری از کامند لاین استفاده می‌کنید. برای استفاده از این ابزار، نام آن را و نام کاربری مورد نظر برای ایجاد حساب جدید وارد می‌کنید. همچنین می‌توانید بین useradd و نام کاربری، گزینه‌های دیگری را نیز اضافه کنید، که در جدول 13.1 خلاصه شده است. دستور useradd از گزینه‌های بیشتری به علاوه گزینه‌هایی که در جدول 13.1 نشان داده شده، پشتیبانی می‌کند؛ برای جزئیات بیشتر می‌توانید به صفحه راهنمای (man page) آن مراجعه کنید.

Table 13.1 Options for useradd

Option name	Option abbreviation	Effect
--comment comment	-C	این گزینه comment field را برای کاربر مشخص می‌کند. ابزارهای گرافیکی به طور عمومی این مورد را به عنوان "full name" توصیف می‌کنند.
--home homedir	-d	این گزینه دایرکتوری home حساب کاربری را مشخص می‌کند. به طور پیشفرض، این دایرکتوری به صورت /home/username تعیین می‌شود.
--expiredate expire-date	-e	این گزینه تاریخی را برای غیرفعال شدن حساب کاربری مشخص می‌کند، که به شکل YYYY-MM-DD اعلام می‌شود. به طور پیشفرض، حساب‌های کاربری غیرمنقضی هستند و تاریخ انقضایی برای آن‌ها تعیین نشده است.
--inactive inactivedays	-f	این گزینه تعداد روزهایی را که پس از انقضای رمز عبور می‌گذرد و حساب کاربری به طور کامل غیرفعال می‌شود، مشخص می‌کند. مقدار -1 این ویژگی را غیرفعال می‌کند و به طور پیشفرض اعمال می‌شود.
--gid defaultgroup	-g	این گزینه نام یا GID پیشفرض کاربر را مشخص می‌کند. به طور پیشفرض، یک گروه جدید با نام کاربر ایجاد می‌شود و به عنوان گروه پیشفرض برای کاربر انتخاب می‌شود.
--groups group[,...]	-G	این گزینه نام‌ها یا شناسه‌های گروه‌هایی را که کاربر عضو آن‌هاست، مشخص می‌کند. می‌توانید چند گروه را با جداکننده کاما مشخص کنید.
--createhome	-m	این گزینه با استفاده از useradd، یک دایرکتوری خانه (directory) برای کاربر ایجاد می‌کند. این گزینه به طور معمول به صورت پیشفرض فعال است.
--skel skeleton-dir	-k	به طور معمول، فایل‌های پیکربندی کاربر پیشفرض از مسیر /etc/skel کپی می‌شوند، اما شما می‌توانید با استفاده از این گزینه، یک دایرکتوری قالب دیگر را مشخص کنید که این گزینه تنها همراه با -m- معتبر است.
None	-M	این گزینه باعث می‌شود که سیستم به طور خودکار دایرکتوری خانه برای کاربر ایجاد نکند.
--shell	-s	این گزینه نام پیشفرض login shell کاربر را مشخص می‌کند.

shell		به طور پیشفرض، این مقدار /bin/bash است.
--uid UID	-u	این گزینه یک حساب کاربری با مقدار UID مشخص را ایجاد می‌کند.
--non-unique	-o	این گزینه امکان استفاده مجدد از یک UID را فراهم می‌کند؛ این گزینه زمانی استفاده می‌شود که حساب کاربری دوم یا بعدی با استفاده از یک UID قبلی ایجاد می‌شود.
--system	-r	این گزینه ایجاد یک حساب کاربری سیستمی را مشخص می‌کند. دستور useradd برای حساب‌های سیستمی دایرکتوری خانه ایجاد نمی‌کند و به آن‌ها مقادیر UID کمتر از 100 اختصاص می‌دهد.
--no-usergroup	-N	این گزینه ایجاد یک گروه برای کاربر را غیرفعال می‌کند.

نکته: در توزیع‌های مبتنی بر Debian و Ubuntu، می‌توانید از ابزار adduser به جای useradd استفاده کنید که به صورت دوستانه‌تر عمل می‌کند. لطفاً توجه داشته باشید که در برخی توزیع‌ها مانند Fedora، دستور adduser، اگر در دسترس باشد، یک جلوه برای ابزار useradd نیست، بلکه یک لینک به آن است.

برخی از این گزینه‌ها به راحتی در دسترس نیستند زمانی که شما از ابزارهای گرافیکی برای ایجاد حساب کاربری استفاده می‌کنید، اما جزئیات متفاوت استفاده از این گزینه‌ها در ابزارهای گرافیکی نیز متفاوت است. در برخی موارد، می‌توان گزینه‌ها را در ابزار گرافیکی پس از ایجاد حساب کاربری تنظیم کرد، اما نه در زمان ایجاد حساب.

یک دستور کامل useradd، شامل تنظیم چند گزینه، به این شکل است:

```
$ sudo useradd -m -c "Hoburn Washburne" -u 1006 hwash
```

[sudo] password for christine:

این مثال یک حساب کاربری با نام کاربری "hwash"، دایرکتوری خانه، میدان توضیحات شامل نام کامل کاربر، و یک UID برابر با 1006 ایجاد می‌کند. در مثال قبلی توجه کنید که دستور sudo برای به دست آوردن دسترسی ادمین (super user) استفاده شده است، که یک الزام برای اجرای موفق این دستور است.

نکته: شما ممکن است بخواهید یک UID را مشخص کنید تا این مقادیر را در انطباق با یکدیگر نگه دارید در کامپیوترهایی که فایل‌ها را با سیستم فایل شبکه (NFS) به اشتراک می‌گذارند، که مالکیت فایل‌ها را از طریق UID تشخیص می‌دهد.

وقتی که یک حساب کاربری با استفاده از دستور useradd ایجاد می‌شود، در حالت قفل قرار می‌گیرد و کاربر قادر به ورود به سیستم نخواهد بود. برای باز کردن قفل حساب، شما باید از دستور passwd استفاده کنید، که در بخش "Modifying Accounts" شرح داده شده است.

نکته: می‌توانید با استفاده از گزینه -p دستور useradd یک رمز عبور را اضافه کنید. با این حال، به دلایل امنیتی، توصیه نمی‌شود از این روش استفاده کنید. بهتر است از دستور passwd استفاده کنید که در بخش بعدی از این فصل شرح داده شده است.

در پشت صحنه، دستور useradd محتوای فایل‌های زیر را تغییر می‌دهد (که در بخش جزئیات آنها در فصل 12 توضیح داده شده است):

/etc/passwd

/etc/shadow

/etc/group

اگر از گزینه -m (یا اگر این گزینه به صورت پیش‌فرض برای توزیع شما باشد) استفاده کنید، برنامه یک دایرکتوری home directory ایجاد می‌کند و فایل‌ها را از /etc/skel به آن مکان کپی می‌کند. ایجاد یک حساب کاربری معمولاً باعث ایجاد یک فایل mail spool می‌شود که در آن ایمیل‌های ورودی کاربر ذخیره می‌شود. (این فایل در بسیاری از سیستم‌های دسکتاپ ممکن است استفاده نشود، اما اگر نرم‌افزار سرور ایمیل را بر روی کامپیوتر اجرا کنید، می‌توانید ببینید که دستور useradd تعداد زیادی تغییرات را در فایل‌ها و دایرکتوری‌های کامپیوتر شما برای ایجاد حساب کاربری انجام می‌دهد.

## Modifying Accounts

همانطور که تازه یاد گرفته‌اید، هنگام ایجاد یک حساب کاربری می‌توانید انواع گزینه‌هایی را مشخص کنید که بر حساب‌ها تأثیر می‌گذارند، مانند اختصاص یک ID خاص به حساب. اما گاهی اوقات نیاز است که بعد از ایجاد حساب، تغییراتی در گزینه‌های حساب اعمال کنید. خوشبختانه، لینوکس ابزارهای گرافیکی و حالت متنی را فراهم می‌آورد که به شما در انجام این کارها کمک می‌کند. قبل از وارد شدن به جزئیات عملیاتی این ابزارها، شما باید بدانید که چه موقعیت‌هایی ممکن است نیاز به اعمال تغییرات در حساب‌ها داشته باشید و همچنین باید بدانید که چگونه بررسی کنید که آیا کاربری در حال حاضر وارد شده است یا خیر.

## Deciding When to Modify Accounts

در یک جهان ایده‌آل، هر بار حساب کاربری خود را به طور کامل و بدون خطأ ایجاد خواهید کرد؛ اما گاهی اوقات این امر ممکن نیست. ممکن است اطلاعاتی که برای ایجاد حساب کاربری بهینه لازم است را نداشته باشید (مانند مدت زمانی که یک کارمند با شرکت خواهد بود)، یا نیازهای شما پس از ایجاد حساب تغییر کند. برخی از علل مشترک برای تغییرات در حساب کاربری عبارتند از (اما به هیچ وجه محدود به زیرند):

- تاریخ انقضای حساب ممکن است نیاز به بروزرسانی داشته باشد، مثلًا اگر قرارداد یک کارمند قراردادی تمدید شود. گاهی اوقات لازم است که یک حساب منقضی را مجددًا فعال کنید.
- شماره‌های ID ممکن است نیاز به همگامسازی با سایر کامپیوترها داشته باشند تا به اشتراک‌گذاری فایل بین کامپیوترها یا به دلایل دیگر کمک کنند.
- دایرکتوری‌های خانه کاربران ممکن است به دلیل افزودن فضای دیسک، باید به یک مکان جدید منتقل شوند.
- یک کاربر ممکن است رمز عبور خود را فراموش کند. مدیر سیستم می‌تواند رمز عبور هر حساب را بدون داشتن رمز عبور اصلی تغییر دهد، بنابراین مدیران سیستم اغلب باید به کاربرانی که یادآوری رمز عبور ناموفقی دارند کمک کنند.

در حالت گرافیکی (GUI)، بسیاری از تغییرات فوق می‌توانند از ابزار مدیریت حساب کاربری GUI توزیع مورد استفاده انجام شوند. اما در text-mode shell، شما باید چندین برنامه مختلف را مسلط شوید تا بتوانید این تغییرات گسترده در حساب‌ها را مدیریت کنید.

## Checking for Logged-in Users

باید به یاد داشته باشید که برخی از تغییرات حساب کاربری ممکن است اگر کاربر در زمان انجام این تغییرات وارد شده باشد، مزاحمت‌آفرین باشند. به ویژه تغییر نام کاربری و دایرکتوری خانه حساب، احتمالاً مشکلاتی ایجاد می‌کند. بنابراین، شما باید فقط زمانی تغییرات حساب را اعمال کنید که کاربر خارج شده باشد. چندین ابزار می‌توانند به شما کمک کنند تا بررسی کنید که کدام کاربر در حال استفاده از کامپیوتر است و در نتیجه مشکلاتی را از پیش بگیرید:

دستور `who`: این ابزار که در فصل 12 شرح داده شده است، یک لیست از کاربرانی که در حال حاضر وارد شده‌اند را نمایش می‌دهد، همراه با برخی جزئیات از جلسات لاگین آن‌ها مانند شناسه‌های ترمینال و تاریخ لاگین.

دستور `w`: این دستور نیز در فصل 12 شرح داده شده است و به طور کلی مشابه `who` است، اما جزئیات متفاوتی ارائه می‌دهد. به طور ویژه، این دستور برنامه‌ای که در هر جلسه در حال اجرا است را شناسایی می‌کند.

نکته: برای یک لیست سریع و ساده از افرادی که در حال استفاده از کامپیوتر هستند، بیشتر توزیع‌ها دستور `users` را نیز ارائه می‌دهند.

دستور `last`: این برنامه یک لیست از جلسات ورود اخیر تولید می‌کند، شامل زمان شروع و پایان آن‌ها، یا یک اعلامیه که کاربر هنوز وارد شده است:

```
$ last
christin pts/2          192.168.0.102      Tue Aug 20 11:46      still
logged in
[...]
reboot    system boot 4.10.0-38-generi Tue Aug 13 10:09 - 11:22
(01:12)
ken        tty4           Tue Aug 6 10:57 - 11:40
(00:43)
devon     tty3           Tue Aug 6 10:57 - 11:40
(00:43)
steph     tty2           Tue Aug 6 10:57 - 11:40
(00:43)
rich      pts/0           192.168.0.102      Tue Aug 6 10:56 - 11:40
(00:43)
christin  tty2           Tue Aug 6 10:34 - 10:57
(00:22)
```

یکی از محدودیت‌های قابل توجه دستور `last` این است که فقط لگین‌های تکست مود را شامل می‌شود. این محدودیت باعث می‌شود که کاربرد آن برای شناسایی کاربرانی که در حال حاضر از کامپیوتر استفاده می‌کنند، نسبتاً محدود باشد، زیرا این کاربران احتمالاً از طریق یک سشن گرافیکی (GUI) وارد شده‌اند.

نکته: دستور `lastb` اطلاعاتی مشابه `last` نمایش می‌دهد، اما فقط تلاش‌های لگین ناموفق را نشان می‌دهد و داده‌های خود را از فایل `/var/log/btmp` می‌گیرد.

دستور `last` داده‌هایی را نمایش می‌دهد که در فایل `/var/log/wtmp` ذخیره می‌شود. باید توجه داشته باشید که برخی از توزیع‌ها این فایل را به طور پیش‌فرض ایجاد نمی‌کنند. برای اطلاعات بیشتر به صفحه راهنمای دستور `last` مراجعه کنید.

## Modifying Accounts Using GUI Tools

مانند افزودن حساب‌های کاربری، روش تغییر حساب‌ها با استفاده از ابزارهای GUI از یک ابزار به دیگری متفاوت است. اکثر ابزارهای گرافیکی گزینه‌های مشابهی ارائه می‌دهند، اگرچه برخی از آن‌ها کامل‌تر از دیگران هستند. در این بخش، یاد خواهید گرفت که چگونه حساب‌ها را با استفاده از ابزار مدیریت کاربران و گروه‌ها در Linux Mint تغییر دهید.

برای انجام این تغییرات، پس از باز کردن ابزار مدیریت کاربران و گروهها، بر روی نام حساب کلیک کنید و سپس روی گزینه "Item To Modify" کلیک کنید. یک نمونه در شکل 13.4 نشان داده شده است.

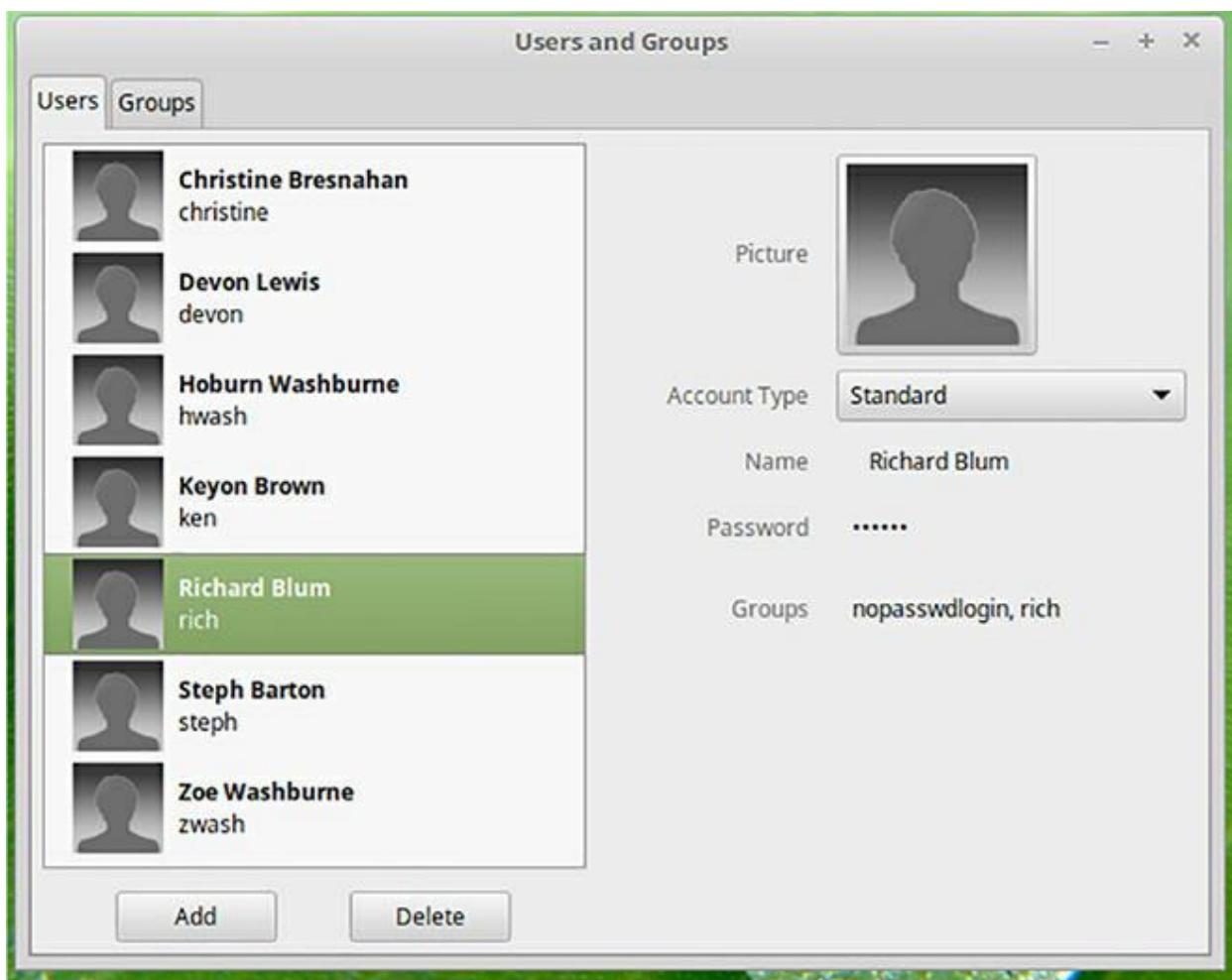


Figure 13.4 The User and Groups Administration utility enables you to edit a few account properties.

این دیالوگ چند ویژگی حساب را ارائه می‌دهد. هر یک از چهار گزینه دسترسی به انواع خاصی از داده‌ها را فراهم می‌کنند:

نوع حساب برای کاربر **Richard Blum** به صورت **Standard** در شکل 13.4 نشان داده شده است. یک حساب استاندارد که همچنین به عنوان حساب کاربری بدون امتیاز شناخته می‌شود، ابتدا در فصل 12 توصیف شد. اگر می‌خواهید این کاربر بتواند در صورت نیاز از طریق دستور `sudo` امتیازات کاربر ادمین را به دست آورد، از منو **Administrator** را انتخاب کنید.

**نکته:** برای استفاده از دستور sudo، حساب ادمین باید عضو گروه wheel باشد، بسته به توزیع شما. افزودن حسابها به گروهها در بخش "Managing Groups" که در ادامه آمده است، پوشش داده شده است.

**Name:** همانطور که در شکل 13.4 نشان داده است، می‌توانید میدان توضیحات حساب (که به عنوان نام شناسایی شده است) را تنظیم کنید. تغییرات نام و در صورت تمایل شرکت شما، ارتقاء موقعیت شغلی نیز می‌تواند در این میدان انجام شود.

**Password:** می‌توانید رمز عبور حساب را با کلیک بر روی فیلد رمز عبور حساب تغییر دهید. یک دیالوگ ظاهر می‌شود، همانطور که قبلاً در شکل 13.3 نشان داده شد. این ابزار به شما اجازه نخواهد داد که رمز عبور حساب را تغییر دهید تا زمانی که آن را به اندازه کافی قوی در نظر بگیرد. در صورت نیاز، می‌توانید با کلیک بر روی پیکان دایره‌ای، یک رمز عبور قوی برای شما تنظیم کند. در این حالت، باید گزینه "Show Password" را تیک بزنید تا بتوانید تنظیم جدید را مشاهده کنید.

**Groups:** برای افزودن این حساب به یک گروه جدید، روی تنظیم گروهها کلیک کنید. یک دیالوگ باز می‌شود که به شما اجازه می‌دهد عضویت‌های گروه اضافی را انتخاب کنید.

**نکته:** اگر می‌خواهید کاربر را به یک گروه کامل‌اً جدید اضافه کنید، ابتدا باید گروه جدید را ایجاد کنید، همانطور که در بخش "Managing Groups" توضیح داده شده است.

کاربران می‌توانند رمز عبور خود را با استفاده از گزینه‌های All در محیط دسکتاپ خود تغییر دهند. به عنوان مثال، در Fedora (Fedora) در ابزار Users، اطلاعات حساب کاربر به صورت خودکار هنگام باز شدن برنامه نمایش داده می‌شود. با کلیک بر روی رمز عبور (که به صورت یک سری نقطه نمایش داده می‌شود)، صفحه تغییر رمز عبور که در شکل 13.5 نشان داده شده است، باز می‌شود. مدیران نیز می‌توانند از این ابزار برای اضافه کردن و تغییر حسابها استفاده کنند.

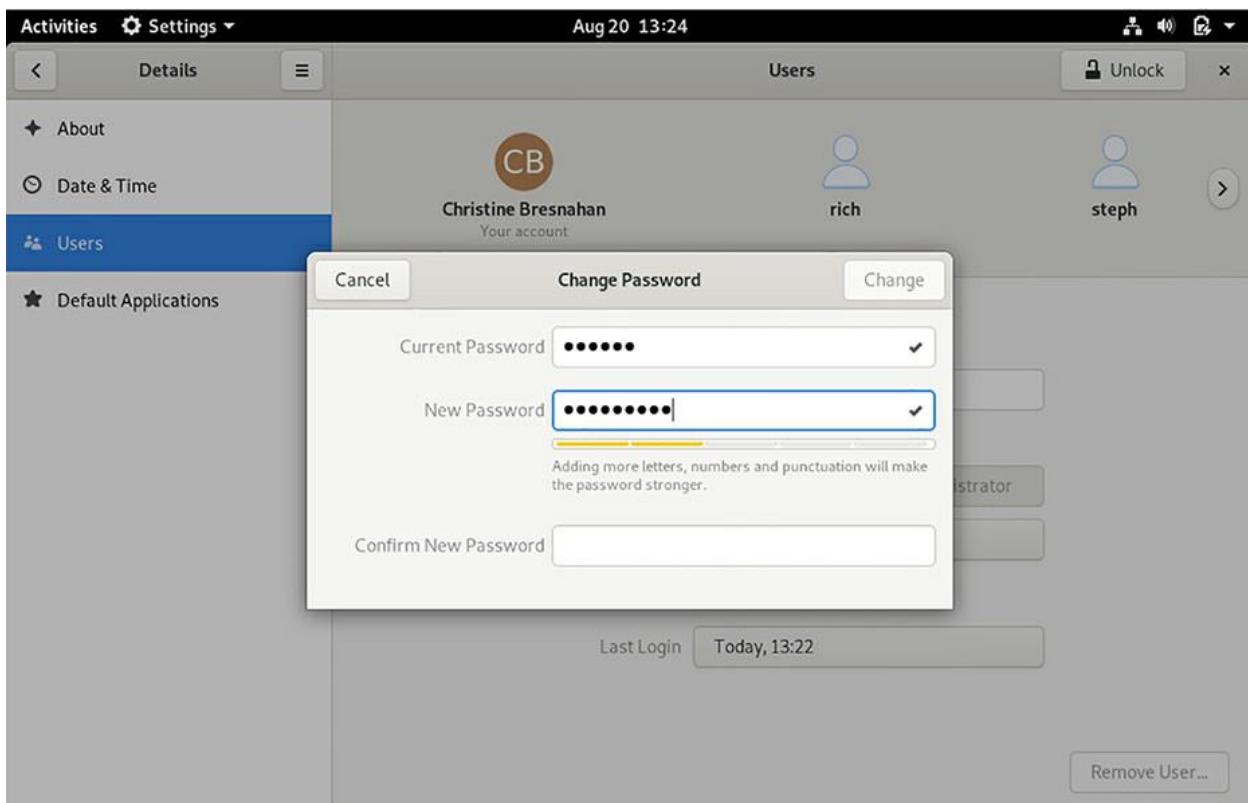


Figure 13.5 The Users utility enables users to change their own passwords.

## Modifying Accounts from the Shell

یکی از تغییرات متدائل حساب، تغییر رمز عبور کاربر است، سه به عنوان بخشی از ایجاد حساب یا به دلیل اینکه کاربر رمز عبور خود را فراموش کرده است. شما می‌توانید این تغییر را با استفاده از برنامه `passwd` انجام دهید. کاربران عادی می‌توانند `passwd` را بنویسند تا رمز عبور خود را تغییر دهند، اما نمی‌توانند رمز عبور حساب‌های کاربری دیگر را تغییر دهند. با این حال، افرادی که امتیازات کاربر ادمین (`super user`) (دارند، می‌توانند نام کاربری را به دستور دهنند تا رمز عبور هر حساب را تغییر دهند:

```
$ sudo passwd hwash
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
$
```

به عنوان تدابیر امنیتی، رمز عبوری که شما تایپ می‌کنید، در حالتی که تایپ می‌کنید، روی صفحه نمایش نمایش داده نمی‌شود. اگر رمزهای عبوری که شما تایپ می‌کنید با هم مطابقت نداشته باشند، برنامه این

تغییر را نپذیرفته و شما را دوباره برای وارد کردن جفت جدیدی از رمزهای عبور دعوت می‌کند. برنامه همچنین قدرت رمز عبور را بررسی می‌کند و ممکن است این رمز عبور جدید را نپذیرفته یا پیام هشداری نمایش دهد اگر تشخیص دهد که رمز عبور به اندازه کافی قوی نیست. لازم به ذکر است که در مثال قبلی دستور sudo استفاده شده است که نشان دهنده این است که فردی که در حال تغییر رمز عبور حساب hwash است، دسترسی به امتیازات کاربر ادمین (super user privileges) دارد، که برای اتمام موققیت‌آمیز این دستور لازم است.

نکته: علاوه بر تنظیم رمز عبور، ابزار `passwd` امکان تنظیم گزینه‌های انقضای رمز عبور و پیرایش را نیز فراهم می‌کند. برای جزئیات بیشتر می‌توانید به صفحه راهنمای `passwd` مراجعه کنید.

شما می‌توانید اکثر تغییرات دیگر حسابها را با استفاده از برنامه `usermod` انجام دهید. این دستور به طرزی شبیه به `useradd` عمل می‌کند، با این تفاوت که به جای ایجاد یک حساب جدید، یک حساب موجود را اصلاح می‌کند. بسیاری از گزینه‌های `usermod` مشابه گزینه‌های `useradd` هستند. جدول 13.2 خلاصه‌ای از مهم‌ترین گزینه‌های `usermod` را نشان می‌دهد.

Table 13.2 Options for usermod

Option name	Option abbreviation	Effect
--append	-a	با استفاده از `groups` یا `G`، این گزینه باعث اضافه شدن گروه‌های مشخص شده به جای جایگزینی آنها می‌شود در مجموعه گروه‌های موجود برای کاربر.
--comment comment	-c	این گزینه، comment field را برای کاربر مشخص می‌کند. ابزارهای GUI عموماً این فیلد را به عنوان "نام کامل" (name) توصیف می‌کنند.
--home home-dir	-d	این گزینه، دایرکتوری home حساب کاربری را مشخص می‌کند. به طور پیش‌فرض، این دایرکتوری به مسیر `/home/username` تنظیم می‌شود، که در آن username نام کاربری انتخابی است.
--expiredate expriedate	-e	این گزینه تاریخی را برای غیرفعال کردن حساب کاربری مشخص می‌کند، که به شکل YYYY-MM-DD بیان می‌شود. به طور پیش‌فرض، حسابی که منقضی نمی‌شود را غیرفعال می‌کند.
--inactive inactivedays	-f	این گزینه تعداد روزهایی را که پس از انقضای رمز عبور، حساب کاربری کاملاً غیرفعال می‌شود، مشخص می‌کند. یک مقدار -1 این ویژگی را غیرفعال می‌کند و به عنوان پیش‌فرض تنظیم شده

		است.
--gid defaultgroup	-g	این گزینه نام یا شناسه گروه پیشفرض کاربر را مشخص می‌کند. به طور پیشفرض، این مقدار یک گروه جدید با نام کاربر ایجاد می‌شود.
--groups group[,...]	-G	این گزینه نامها یا شناسه‌های گروه‌هایی که کاربر عضو آنها است را مشخص می‌کند. می‌توان بیش از یک گروه را با جدا کردن آنها با کاما مشخص کرد.
--login username	-l	این گزینه نام کاربری حساب را به مقدار مشخص شده تغییر می‌دهد.
--lock	-L	این گزینه رمز عبور حساب را قفل می‌کند و اجازه ورود به حساب را نمی‌دهد.
--move-home	-m	وقتی این گزینه به همراه (-d) در `usermod` استفاده می‌شود، `usermod` دایرکتوری خانه موجود کاربر را به مکان جدید منتقل می‌کند.
--shell shell	-s	این گزینه نام شل پیشفرض کاربر را مشخص می‌کند.
--uid UID	-u	این گزینه شماره شناسه کاربری (UID) حساب را به مقدار مشخص شده تغییر می‌دهد.
--unlock	-U	این گزینه رمز عبور قفل شده حساب کاربری را باز می‌کند، به این ترتیب که کاربر دوباره می‌تواند وارد حساب کاربری خود شود.

به عنوان مثال، از `usermod` به شکل زیر استفاده کنید:

```
$ sudo usermod -u 1072 -m -d /home2/hwash hwash
```

این دستور سه تغییر به حساب کاربری hwash اعمال می‌کند:

- شماره شناسه کاربری (UID) را به مقدار 1072 تغییر می‌دهد.
- دایرکتوری خانه حساب کاربری را به مسیر /home2/hwash /تغییر می‌دهد.
- محتويات دایرکتوری خانه اصلی حساب کاربری را به مکان جدید آن منتقل می‌کند.

شما ممکن است همچین دستوری را صادر کنید اگر در حال مهاجرت حساب‌های کاربری به سرور NFS با مسیر 2home/ باشید. چنین تغییری ممکن است نیازمند یک مکان جدید برای دایرکتوری خانه و تغییر شماره UID باشد تا با مقدار استفاده شده در سرور NFS همخوانی داشته باشد.

نکته: هنگامی که تغییراتی در مقدار UID اعمال می‌کنید، باید مراقب باشید، زیرا usermod اغلب مقادیر UID فایل‌هایی را که در مکان‌های معمول مانند دایرکتوری خانه کاربر و فایل‌های ایمیل قرار دارند، تغییر می‌دهد، اما ممکن است فایل‌های کاربر در مکان‌های ناشناخته دیگر را از دست بدهد.

اگر نیاز به اعمال تغییرات گروه دارید که شامل افزودن گروه‌های جدید است، برای اطلاعات بیشتر در این زمینه به بخش "Managing Groups" مراجعه کنید.

## Deleting Accounts

حذف حساب‌های کاربری گاهی اوقات به اندازه اضافه کردن یا اصلاح آنها مهم است. حساب‌هایی که استفاده نمی‌شوند ممکن است توسط صاحبان قبلی خود یا افراد دیگری که ممکن است بتوانند با استفاده از گذر واژه‌های ضعیف به حساب‌ها نفوذ کنند، سوء استفاده شوند. بنابراین، شما باید به طور دوره‌ای حساب‌های استفاده نشده را حذف کنید. قبل از انجام این کار، باید بفهمید که وقتی یک حساب را حذف می‌کنید، چه اتفاقی می‌افتد و تصمیم دقیقی برای انجام آن بگیرید، تا با حذف یک حساب به شکل نامناسب مشکلاتی ایجاد نکنید. با داشتن این دانش، شما می‌توانید با استفاده از ابزارهای GUI یا تکست مود، حساب‌ها را حذف کنید.

## Avoiding Account Deletion Pitfalls

حذف یک حساب کاربری ممکن است به نظر به اندازه کافی ساده باشد، اما یک اشتباه می‌تواند مشکلاتی را به وجود آورد، یا به طور فوری یا در آینده. به جز مشکلات آشکار مانند حذف اشتباه حساب، دو عامل زیر را نیز در نظر بگیرید:

**حفظ فایل‌های کاربر:** فایل‌های کاربران ممکن است بسیار ارزشمند باشند، چه برای خود کاربران و چه برای سازمانی که از کامپیوتر مالکیت می‌کند. شما باید سیاست‌های نگهداری فایل شرکت خود را بررسی کنید تا در نظر داشته باشید که آیا باید دایرکتوری خانه کاربر را حذف کنید یا اقدامی دیگری مانند انتقال آن به دایرکتوری خانه یک کاربر دیگر و تغییر دسترسی‌های فایل‌های در آن را انجام دهید. همین مساله برای صفت ایمیل کاربر (معمولًاً در مسیر /var/spool/mail/username که در آن username نام کاربری حساب است) نیز صادق است.

نکته: در نظر داشته باشید که دایرکتوری خانه حساب کاربری حذف شده را به یک رسانه پشتیبانی بلند مدت فشرده کنید. این استراتژی به شما اجازه می‌دهد تا در صورتی که فایل‌ها در آینده ارزشمند شوند، قادر باشید آنها را بازیابی کنید.

**UID and GID Reuse:** هنگامی که یک حساب کاربری حذف می‌شود، شماره‌ی شناسه کاربری (UID) و شناسه گروه (GID) حساب مورد نظر برای استفاده مجدد در دسترس قرار می‌گیرد. در بسیاری از موارد، این شماره‌ها بلافصله مجدد استفاده نمی‌شوند، زیرا اکثر توزیع‌های لینوکس ارزش‌دهی این شماره‌ها را بر اساس بالاترین مقدار موجود انجام می‌دهند. اگر یک حساب کاربری با شماره‌ی UID یا GID کمتر از بالاترین مقدار فعلی حذف شود، این شماره‌ها تا زمانی که حساب‌های میانی نیز حذف شوند، مجدد استفاده نمی‌شوند. با

این حال، در صورت استفاده مجدد از یک UID، هر فایلی که پیشتر توسط کاربر قدیمی مالک آن بوده است، به طور ناگهانی به مالکیت کاربر جدید منتقل می‌شود. این ممکن است باعث ایجاد ابهام درباره‌ی اینکه کدام فایل توسط کدام کاربر ایجاد شده است. در برخی موارد، این می‌تواند باعث شک و تردید درباره‌ی نامناسب بودن رفتار جدید کاربر شود (اگر فایلهای قدیمی اطلاعاتی را شامل شوند که کاربر جدید نباید به آن دسترسی داشته باشد یا اگر فایل‌ها در دایرکتوری‌هایی باشند که کاربر جدید باید به آنها دسترسی نداشته باشد).

برای جلوگیری از هرگونه ابهام یا ادعاهای نادرست نسبت به رفتار یا مشکلات ناشی از استفاده مجدد از UID یا GID، می‌توانید از دستور find (که در بخش جزئیات آن در فصل 8، "Searching, Extracting, and "Archiving Data پوشش داده شده است) برای یافتن همه‌ی فایلهایی با مقادیر خاص UID یا GID استفاده کنید. باید از گزینه‌های uid- و gid- استفاده کنید، همانطور که در زیر نشان داده شده است:

```
$ sudo find / -uid 1004
```

می‌توانید این دستور را بدون استفاده از امتیازهای مدیریتی فوق العاده صادر کنید، اما این ممکن است با خطاهای روبرو شود و ممکن است برخی از فایل‌ها را از دست بدهد. بنابراین، بهتر است از find به این شکل با امتیازهای مدیریتی فوق العاده استفاده کنید.

مثال قبلی ما تمام فایلهای موجود در کامپیوتر با UID 1004 را پیدا می‌کند. (جستجو برای GID به همان روش انجام می‌شود، اما با استفاده از گزینه gid-) سپس می‌توانید مالکیت این فایل‌ها را با استفاده از دستور chown (که در فصل 14 پوشش داده شده است) مجددأ تعیین کنید یا آنها را حذف کنید. به طور معمول، شما این دستور را تنها پس از حذف یا تغییر مالکیت دایرکتوری خانه‌ی کاربر خواهید داد، زیرا این دایرکتوری احتمالاً شامل خیلی از فایلهای مطابق با این شرایط خواهد بود.

## Deleting Accounts Using GUI Tools

به عنوان سایر وظایف مدیریت حساب کاربری، استفاده از رابط گرافیکی بسیار بدون هیچ مشکلی است، اما جزئیات می‌توانند بین توزیع‌های مختلف متفاوت باشند. به عنوان مثال، برای حذف یک حساب کاربری از ابزار مدیریت کاربران و گروه‌ها در Linux Mint، پس از باز کردن این ابزار، بر روی حساب کاربری کلیک کرده و سپس بر روی دکمه Delete کلیک کنید. نتیجه این کار یک پنجره تأیید مشابه با آنچه در شکل 13.6 نشان داده شده است، خواهد بود. اگر از این عمل اطمینان دارید، بر روی دکمه Yes کلیک کنید. حساب کاربری بلاfacle حذف خواهد شد.

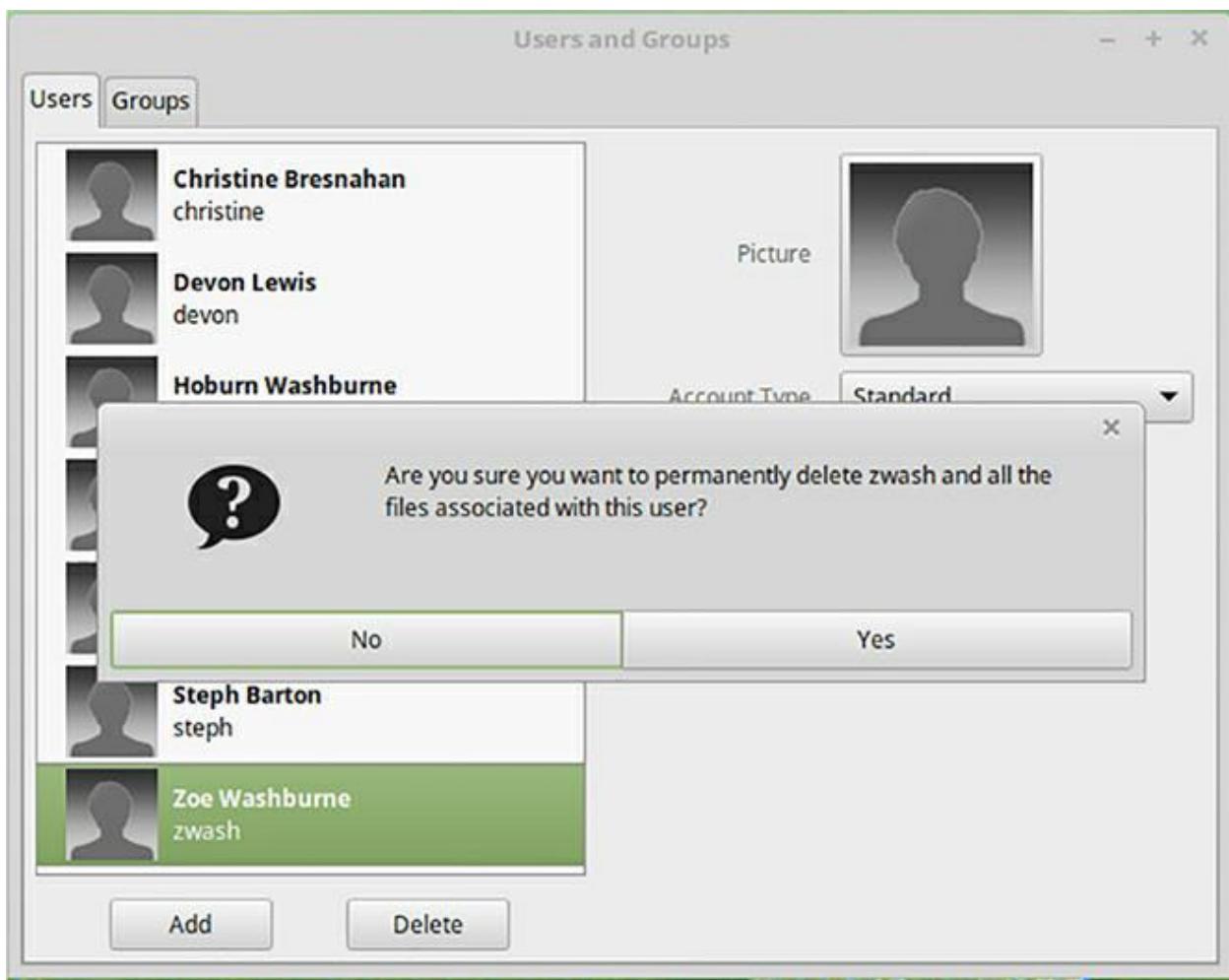


Figure 13.6 When you delete an account with a GUI utility, you are typically asked to confirm the action.

اگر کاربر در حال حاضر وارد شده باشد، ابزار معمولاً در این باره ارور خواهد داد. احتمالاً شما همچنان می‌توانید حساب کاربری را حذف کنید، اما کاربر به طور فوری خارج نخواهد شد.

## Deleting Accounts from the Shell

دستور `userdel` از یک text-mode shell حساب‌های کاربری را حذف می‌کند. در شکل ساده‌ترین آن، با استفاده از امتیازهای مدیریتی فوق‌العاده، شما یک نام کاربری را به آن انتقال می‌دهید و نه بیشتر:

```
$ sudo userdel hwash
```

```
[sudo] password for christine:
```

این برنامه شما را برای تأیید حذف نمی‌طلبد؛ به طور مستقیم حساب کاربری را حذف می‌کند. با این حال، به طور پیش فرض، دایرکتوری خانه کاربر را حذف نمی‌کند. برای این کار، به آن گزینه `-remove` یا `-r` را انتقال دهید.

اگر کاربر در حال حاضر وارد شده باشد، `userdel` شما را از این موضوع مطلع می‌کند و هیچ کاری انجام نمی‌دهد. می‌توانید به آن گزینه `-f` را انتقال دهید تا حساب کاربری را حذف کنید، حتی اگر در حال استفاده باشد. برای همچنین اجبار حذف حساب و پاک کردن فایل‌های کاربر، می‌توانید هر دو گزینه را انتقال دهید:

```
$ sudo userdel -rf zwash
```

```
[sudo] password for christine:
```

```
userdel: user zwash is currently used by process 4800
```

```
$
```

برنامه هنوز به ارور دادن به کاربر برای "used by a process" (وارد سیستم شده است) ادامه می‌دهد، اما با این حال، حساب و فایل‌ها را همچنان حذف می‌کند.

## Managing Groups

در بسیاری از جنبه‌ها، گروه‌ها با حساب‌های کاربری قابل مقایسه هستند. آن‌ها در فایل‌های مشابهی تعریف می‌شوند و با ابزارهای مشابه مدیریت می‌شوند. گروه‌ها همچنین با حساب‌ها مرتبط هستند به این معنی که حساب‌ها شامل تعریف‌های گروهی می‌شوند. تا به این لحظه، فرض شده است که شما از گروه‌های استاندارد یا گروه‌هایی که به عنوان بخشی از ایجاد حساب تعریف شده‌اند، استفاده می‌کنید. با این حال، گاهی اوقات نیاز دارید که برای اهداف خاصی مانند استفاده از استراتژی گروه پروژه، گروه‌ها را ایجاد، حذف یا اصلاح کنید. مشابه مدیریت حساب‌ها، می‌توانید از ابزارهای GUI یا تکست مود استفاده کنید.

## Managing Groups Using GUI Tools

ابزارهای GUI بسیاری برای نگهداری حساب، مانند ابزار مدیریت کاربران و گروه‌های لینوکس می‌یعنی، ابزارهای مدیریت گروه ارائه می‌دهند که مشابه ابزارهای مدیریت کاربرانی هستند که در این فصل توضیح داده شده‌اند. با اشاره به شکل 13.1، شما می‌توانید ببینید که پنجره مدیریت کاربران و گروه‌ها شامل دو تب کاربران و گروه‌ها است. برای مدیریت گروه‌ها، باید بر روی تب گروه‌ها کلیک کنید. این کار باعث نمایشی مشابه شکل 13.7 می‌شود.

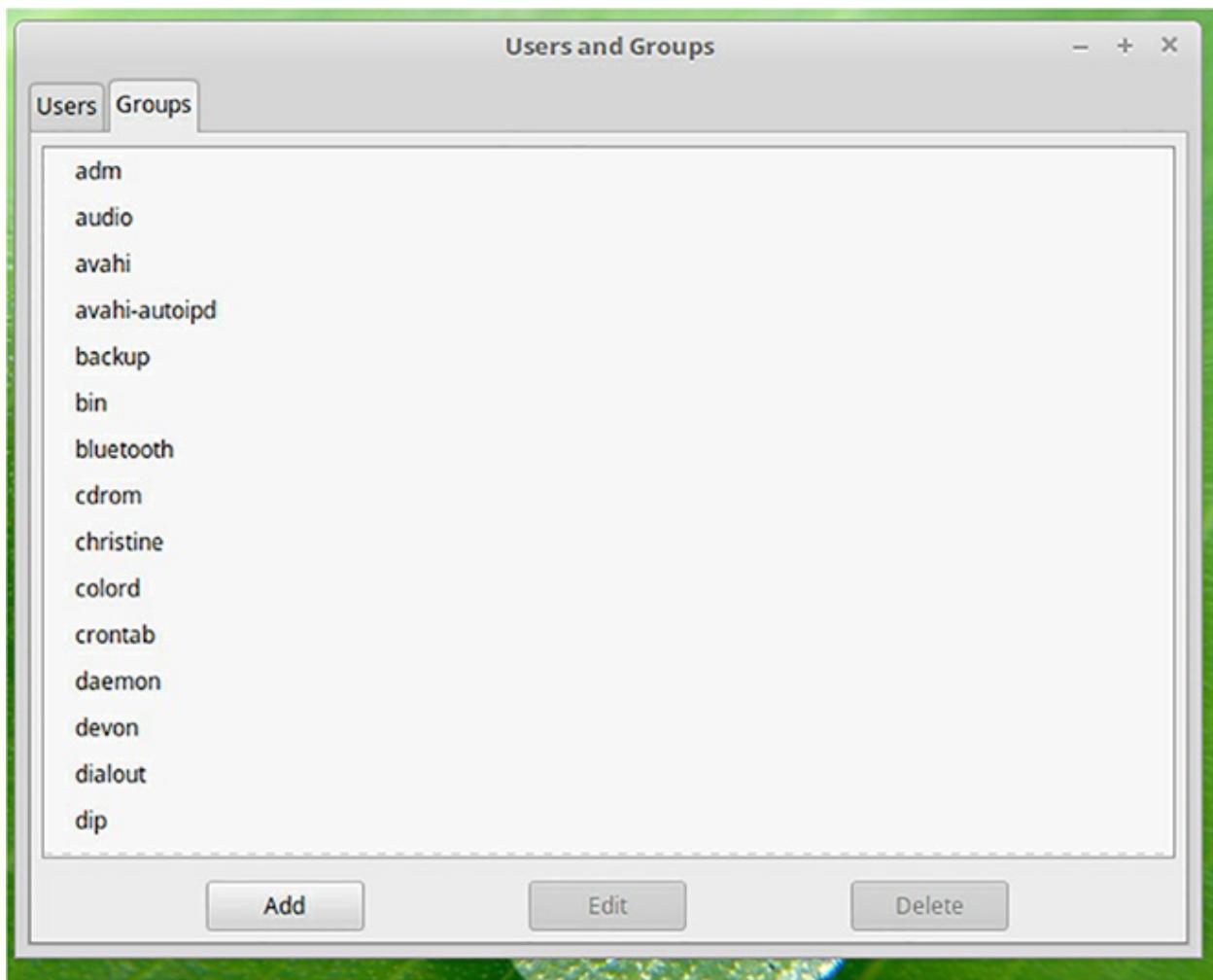


Figure 13.7 The User and Groups Administration utility enables you to manage groups as well as users.

می‌توانید گروه‌ها را به روشی مشابه با افزودن، اصلاح و حذف حساب‌ها مدیریت کنید، اگرچه تعداد گزینه‌های موجود ممکن است کمتر باشد. گروه‌ها دارای دایرکتوری خانه، فیلد توضیحات، شل‌های ورودی و موارد مشابه نیستند. البته ممکن است کاربران خاصی را از ابتدا به عنوان اعضای گروه جدید خود بخواهید. برای انجام این کار، مراحل زیر را دنبال کنید:

1. با کلیک بر روی دکمه افزودن در تب گروه‌ها، گروهی را ایجاد کنید.
2. نام گروه را در دیالوگ باکس نمایش داده شده مشخص کنید.
3. با کلیک بر روی OK، گروه را اضافه کنید.
4. اعضای گروه را اضافه کنید، با انتخاب تب کاربران، انتخاب نام حساب کاربری، و کلیک بر روی گروه‌های فعلی در فیلد گروه‌ها برای باز کردن یک دیالوگ.
5. با انتخاب باکس نام گروه جدید در دیالوگ، آن حساب را به عنوان عضوی از گروه تعیین کنید، مانند شکل 13.8، و روی OK کلیک کنید.

نکته: شما نیازی ندارید که حساب را فقط به یک گروه در هر زمان اضافه کنید. می‌توانید با کلیک بر روی چند گروه در دیالوگ، حساب را به چند گروه همزمان اضافه کنید.

می‌توانید به طور جایگزین با تغییر عضویت گروهی هر کاربر به طور جداگانه، به مدیریت عضویت گروه پردازید، که در ادامه در "Managing Groups from the Shell". شرح داده شده است.

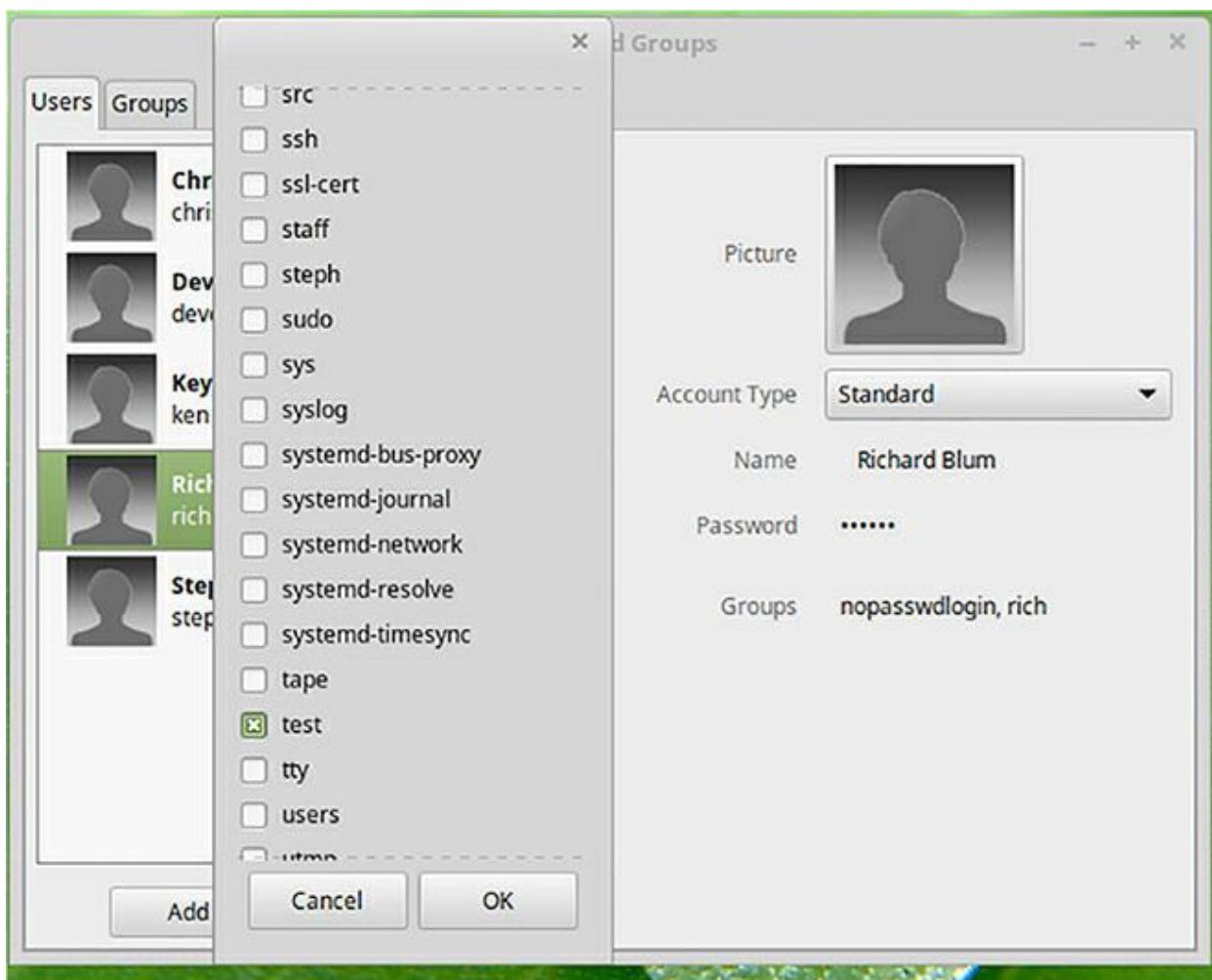


Figure 13.8 You can add users to a group after creating the group.



## Real World Scenario

### The wheel Group

Most Linux توزیع‌های لینوکس در همیشه چندین گروه را به طور پیش‌فرض فراهم می‌کنند. یکی از این گروه‌ها، wheel (در برخی از توزیع‌ها)، به ویژه برای مدیریت سیستم اهمیت دارد. اعضای wheel دارای امتیازات ویژه مدیریتی هستند، مانند حق استفاده از دستور sudo. نام گروه از اصطلاح خودرو، که به شخص مهم اشاره دارد، برمی‌گردد.

همه توزیع‌ها این گروه را فراهم نمی‌کنند - به عنوان مثال، لینوکس می‌بیند از گروه sudo برای اعطای دسترسی به دستور sudo استفاده می‌کند. شما می‌توانید با تایپ grep wheel /etc/group بزرگی کنید که آیا گروه wheel در توزیع شما وجود دارد یا خیر. اگر وجود دارد، می‌توانید با تایپ grep wheel /etc/sudoers و جستجو برای نتایج مشابه ALL=(ALL) ALL% ببینید که عضویت در این گروه دسترسی به دستور sudo را فراهم می‌کند.

برخی از توزیع‌ها به شما امکان افزودن حساب کاربری اصلی خود به گروه wheel را هنگام نصب سیستم عامل می‌دهند. برای بررسی آیا این امکان وجود دارد یا خیر، باید به عنوان "گروه مدیران" یا اصطلاح مشابه مورد سوال نصب کننده استفاده شود.

## Managing Groups from the Shell

شما می‌توانید با استفاده از دستور `groupadd` از کامند لاین گروه‌ها را ایجاد کنید، که مشابه به کارکرد `useradd` برای کاربران است، اما با مجموعه کوچکتری از گزینه‌ها، که مهم‌ترین آن‌ها در جدول 13.3 آورده شده‌اند. برای اطلاعات بیشتر در مورد گزینه‌های پنهان، به صفحه راهنمای دستور برنامه مراجعه کنید.

Table 13.3 Options for groupadd

Option name	Option abbreviation	Effect
--gid GID	-g	تعیین یک GID خاص. اگر آن را حذف کنید، `groupadd` از `groupadd` بعدی در دسترس استفاده می‌کند.
--system	-r	دستور `groupadd` را دستور گروه سیستمی ایجاد می‌کند که دارای یک GID کمتر از 500 یا 1000 است، بسته به توزیع. گروه‌های غیرسیستمی معمولاً به عنوان گروه‌های خصوصی کاربر استفاده می‌شوند.
--force	-f	به طور معمول، اگر سعی کنید یک گروه را که قبلاً وجود دارد، ایجاد کنید، دستور `groupadd` یک پیام خطابازمی‌گرداند. این پارامتر این

پیام خط را قابلیت سرکوب می‌کند.

استفاده از دستور groupadd، از جمله تنظیم یک گزینه، به شکل زیر است:

```
# groupadd -g 1001 consultants
```

این مثال یک گروه با شناسه گروهی (GID) برابر با 1001 و نام consultants ایجاد می‌کند. توجه کنید که در مثال قبلی نشانه # نمایش داده شده است، که نشان می‌دهد که حساب روت برای به دست آوردن دسترسی به super user برای اجرای موفق این دستور استفاده شده است.

پس از اضافه شدن گروه، اعضای جدید می‌توانند با استفاده از دستور usermod و دسترسی super user، به گروه اضافه شوند، به شرح زیر:

```
# usermod -aG consultants rich
```

```
##
```

```
groups rich
```

```
rich: users consultants
```

دستورات G-a- در کنار هم استفاده شده‌اند تا حساب rich را به گروه جدید consultants اضافه کنند. اگر از گزینه a- استفاده نمی‌شد، در برخی توزیع‌ها حساب rich از اعضای فعلی گروه‌هاییش حذف می‌شد و تنها عضو گروه consultants می‌ماند. بنابراین، استفاده از G-a- به طور پیش‌فرض معقول است.

نکته: می‌توانید گروه اصلی یک حساب کاربری را بررسی کنید. با دسترسی مدیر سیستم، دستور `id -gn `username` را در کامند لاین تایپ کنید. گروه اصلی نمایش داده می‌شود.

یک عادت خوب دیگر این است که با استفاده از دستور `groups` بررسی کنید که آیا اصلاحات با موفقیت انجام شده یا نه. دستور `groups` تمامی عضویت‌های گروه را برای حساب کاربری مشخص شده نمایش می‌دهد.

نکته: به منظور اصلاح خود گروه، می‌توانید از دستور groupmod استفاده کنید. گزینه‌های gid- یا -g و non-new- یا -o از جدول 13.3 می‌توانند با این دستور استفاده شوند، همچنین می‌توانید از گزینه -unique- یا -n name name استفاده کنید که نام گروه را تغییر می‌دهد.

حذف گروه‌ها از طریق کامند لاین از دستور groupdel استفاده می‌کند، که نام گروه را به عنوان یک گزینه .consultants groupdel consultants برای حذف گروه می‌پذیرد، مانند

# CHAPTER 14

## Setting Ownership and Permissions

به عنوان یک سیستم عامل چندکاربره، لینوکس ابزارهایی را فراهم می‌کند تا به شما کمک کند تا فایل‌های خود را در برابر دسترسی‌های نامطلوب محافظت کنید؛ در نهایت، شما نمی‌خواهید که یک کاربر دیگر به طور تصادفی (یا عمدی) فایل‌های شخصی شما را بخواند یا حتی حذف کند! لینوکس این وظایف را از طریق دو ویژگی اصلی فایل‌ها و دایرکتوری‌ها مدیریت می‌کند: مالکیت و دسترسی‌ها. هر فایل دارای یک مالک (یعنی یک حساب کاربری که به آن متصل است) و یک گروه مرتبط است. سه مجموعه دسترسی تعیین می‌کنند که مالک فایل، اعضای گروه فایل و سایر کاربران می‌توانند با فایل انجام دهند. بنابراین، مالکیت و دسترسی‌ها به یکدیگر پیوند خورده‌اند، با این حال شما از دستورات متنی متفاوت برای مدیریت آن‌ها استفاده می‌کنید. (ابزارهای گرافیکی معمولاً این دو را ترکیب می‌کنند، همانطور که در این فصل شرح داده شده است).

### Setting Ownership

مدل امنیتی لینوکس بر اساس مدل امنیتی یونیکس است که به عنوان یک سیستم عامل چندکاربره طراحی شده بود. این مدل امنیتی بنابراین فرض می‌کند که چندین کاربر بر روی کامپیوتر حضور دارند و وسائل لازم برای ارتباط فایل‌های فردی با کاربرانی که آن‌ها را ایجاد کرده‌اند فراهم می‌کند؛ یعنی فایل‌ها مالک دارند. شما باید این مفهوم را به طور کامل درک کنید و با آن دانش، می‌توانید از فایل‌های خود محافظت کنید، با استفاده از یک مدیر فایل گرافیکی یا دستورات متنی.

مالکیت همچنین به برنامه‌های در حال اجرا (یعنی فرایندها) اعمال می‌شود. بیشتر برنامه‌هایی که اجرا می‌کنید به حساب کاربری که برای راهاندازی آن‌ها استفاده کرده‌اید مرتبط هستند. این هویت، همراه با مالکیت و دسترسی‌های فایل، تعیین می‌کند که آیا یک برنامه می‌تواند یک فایل را تغییر دهد یا نه.

### Understanding Ownership

فصل 12، "Creating Users and Groups"، فصل 13، "Understanding Basic Security" و فصل 13، "Creating Users and Groups" سیستم حساب‌های لینوکس را توضیح دادند. این حساب‌ها اساس مالکیت فایل‌ها هستند. به طور خاص، هر فایل یک مالک دارد—یک حساب که با آن مرتبط است. شماره UID فایل را با یک مالک مرتبط می‌کند، در حالی که شماره GID فایل را با یک گروه مرتبط می‌کند.

همان‌طور که بعداً در بخش "Setting Permissions" توضیح داده شده است، شما دسترسی به فایل را از طریق دسترسی‌هایی که به طور مستقل برای مالک فایل، گروه فایل، و سایر کاربران کامپیوتر تنظیم می‌کنید، کنترل می‌کنید. به عنوان روت، می‌توانید مالک و گروه هر فایل را تغییر دهید. مالک فایل نیز می‌تواند گروه فایل را تغییر دهد، اما تنها به گروهی که خود به آن تعلق دارد.

اصول مالکیت مشابه برای دایرکتوری‌ها نیز اعمال می‌شود: دایرکتوری‌ها دارای مالک و گروه هستند. آنها می‌توانند توسط روت یا به میزان محدودتری توسط مالک دایرکتوری تغییر یابند.

## Setting Ownership in a File Manager

همان‌طور که در فصل 4، "Using Common Linux Programs"، توضیح داده شد، می‌توانید فایل‌ها را با یک مدیر فایل مدیریت کنید. احتمالاً با مدیران فایل در ویندوز یا macOS آشنا هستید. با این حال، مالکیت و دسترسی‌های لینوکس با ویندوز متفاوت است، بنابراین ممکن است بخواهید بدانید چگونه ویژگی‌های مالکیت را با استفاده از یک مدیر فایل لینوکس بررسی کرده و شاید تغییر دهید. همان‌طور که در فصل 4 ذکر شد، شما می‌توانید از چندین مدیر فایل در لینوکس استفاده کنید. بیشتر آنها در کلیات مشابه هستند، اما در جزئیات تفاوت دارند. در این بخش از فایل منیجر GNOME Files به عنوان مثال استفاده خواهیم کرد.

اگر می‌خواهید مالک فایل را تغییر دهید، باید فایل‌ها را به عنوان روت اجرا کنید، اما می‌توانید گروه فایل را به هر گروهی که به آن تعلق داردید به عنوان یک کاربر عادی تغییر دهید. روش انجام این کار به عنوان روت به شرح زیر است:

1. یک پنجره ترمینال باز کنید.
2. در پنجره ترمینال، عبارت `su` را برای کسب دسترسی‌های روت تایپ کنید.

نکته: برخی از توزیع‌های لینوکس به شما اجازه نمی‌دهند از دستور `su` برای کسب دسترسی‌های روت استفاده کنید. به عنوان مثال، اگر از نسخه GNOME اوبونتو استفاده می‌کنید، ممکن است به جای آن نیاز باشد از `sudo` برای اجرای GNOME Files از کامند لاین استفاده کنید.

3. در پنجره ترمینال، `nautilus` را تایپ کنید تا GNOME Files اجرا شود (برنامه GNOME Files در نسخه‌های قبلی GNOME به نام Nautilus شناخته می‌شد و این نام همچنان باقی مانده است). می‌توانید به صورت اختیاری مسیر دایرکتوری که می‌خواهید Files از آنجا شروع به کار کند را نیز وارد کنید. اگر مسیری وارد نکنید، برنامه با نمایش محتويات دایرکتوری `/root/` آغاز خواهد شد.

نکته: دایرکتوری `/root/` دایرکتوری خانه حساب کاربری روت است.

4. فایل مورد نظر که مالکیت آن را می‌خواهید تغییر دهید را پیدا کرده و بر روی آن راست کلیک کنید.
5. در منوی باز شده، گزینه "Properties" را انتخاب کنید. نتیجه یک دیالوگ "Properties" خواهد بود.
6. در دیالوگ "Properties"، بر روی زبانه "Permissions" کلیک کنید. نتیجه شبیه شکل 14.1 خواهد بود.
7. برای تغییر مالک فایل، مالک جدید را در فیلد "Owner" انتخاب کنید. این عمل تنها در صورتی ممکن است که GNOME Files را به عنوان root اجرا کرده باشید.

8. برای تغییر گروه فایل، گروه جدید را در فیلد "Group" انتخاب کنید. اگر GNOME Files را به عنوان کاربر عادی اجرا کنید، میتوانید هر گروهی که به آن تعلق دارد را انتخاب کنید، اما اگر GNOME Files را به عنوان روت اجرا کنید، میتوانید هر گروهی را انتخاب کنید.
9. هنگامی که ویژگی‌های مورد نظر خود را تغییر دادید، بر روی X در نوار عنوان کلیک کنید تا پنجره بسته شود.

اگر میخواهید گروه یک فایل را تغییر دهید اما مالک آن را تغییر ندهید، و اگر عضو گروه هدف هستید، میتوانید GNOME Files را به عنوان یک کاربر عادی راهاندازی کنید. سپس میتوانید از مرحله 4 در رویه قبلی شروع کنید.

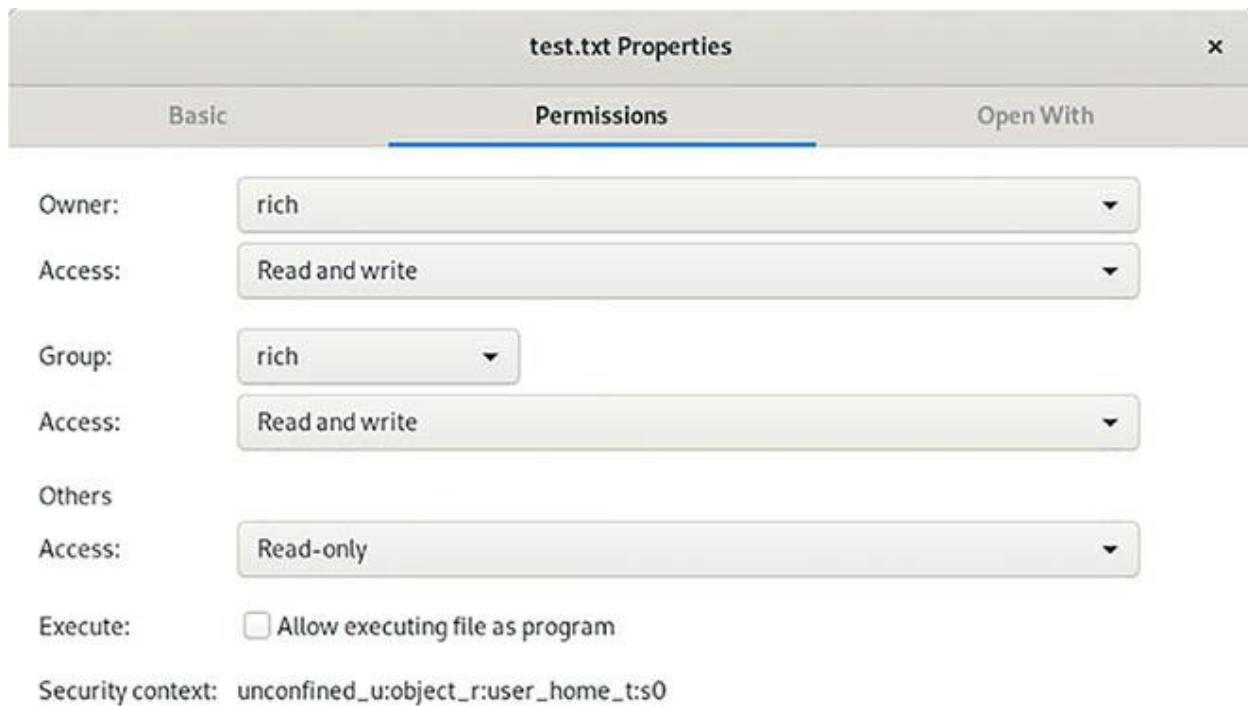


Figure 14.1 Linux file managers give you access to the file's ownership and permission metadata.

شما باید در اجرای GNOME Files به عنوان root بسیار محظوظ باشید. اگر فراموش کنید که این برنامه را به عنوان روت اجرا کرده‌اید، میتوانید به راحتی فایل‌های جدیدی به عنوان روت ایجاد کنید که برای انجام هر گونه تغییری در مالکیت یا مجوزهای فایل در آینده نیاز به دسترسی روت دارند. همچنین به سادگی میتوانید به طور تصادفی فایل‌های سیستم حیاتی را به عنوان روت حذف کنید که نمیتوانستید به عنوان یک کاربر عادی حذف کنید. به همین دلایل، توصیه میکنم از یک شل تکست مود برای تنظیم مالکیت فایل استفاده کنید. تغییر در اعلان باعث می‌شود راحت‌تر متوجه شوید که به عنوان روت اجرا میکنید، و اگر به استفاده از GUI عادت دارید، احتمال کمتری وجود دارد که برنامه‌های اضافی را به عنوان روت از یک شل تکست مود اجرا کنید تا از GNOME Files.

## Setting Ownership in a Shell

دستور برای تغییر مالکیت یک فایل به روش متنی ترجیحی، chown است. در ساده‌ترین شکل آن، نام فایل را پس از آن نام کاربری قرار می‌دهید:

نکته: نام دستور chown مخفف عبارت "change owner" است، به معنای تغییر مالک فایل.

```
# chown rich targetfile.odf
```

این مثال مالکیت فایل targetfile.odf را به کاربر rich اختصاص می‌دهد. می‌توانید با یک دستور فایل را به عنوان مالک اصلی و گروه آن تغییر دهید که این کار با جدا کردن نام مالک و گروه با دونقطه (: ) یا یک نقطه (.) انجام می‌شود:

```
# chown bob:users targetfile.odf
```

این مثال مالکیت فایل targetfile.odf را به کاربر bob اختصاص می‌دهد و فایل را با گروه کاربران مرتبط می‌کند. برای تغییر گروه بدون تغییر مالک، می‌توانید مالک را حذف کنید و فقط نام گروه را باقی بگذارید که با استفاده از دونقطه (: ) انجام می‌شود:

```
$ chown :users targetfile.odf
```

به عنوان جایگزین، می‌توانید از دستور chgrp استفاده کنید که به همان شکل عمل می‌کند اما فقط گروه را تغییر می‌دهد و نیازی به استفاده از دونقطه قبل از نام گروه ندارد:

```
$ chgrp users targetfile.odf
```

توجه داشته باشید که دستوراتی که برای تغییر مالکیت استفاده می‌شوند نیاز به دسترسی root دارند، در حالی که شما به عنوان کاربر معمولی می‌توانید گروه را تغییر دهید - اما فقط اگر فایل متعلق به شما باشد و عضو گروه مورد نظر باشید.

دستورات chown و chgrp هر دو از چندین گزینه پشتیبانی می‌کنند. مفیدترین آنها -R (یا recursive) است که باعث تغییر مالکیت تمام فایل‌ها در یک درخت دایرکتوری می‌شود. به عنوان مثال، فرض کنید که کاربر christine از شرکت خارج شده است و یک کارمند موجود، rich، باید به فایل‌های او او دسترسی داشته باشد. اگر دایرکتوری خانه christine /home/christine بود، می‌توانید این دستور را تایپ کنید:

```
# chown -R rich /home/christine
```

این دستور مالکیت دایرکتوری /home/christine را به کاربر rich می‌دهد؛ تمام فایل‌های موجود در دایرکتوری /home/christine، شامل تمام زیردایرکتوری‌ها و فایل‌های درون آن، به rich تعلق می‌گیرد. برای تسهیل بیشتر در انتقال، ممکن است بخواهید دایرکتوری خانه پیشین christine را به داخل دایرکتوری خانه rich منتقل کنید.



## Real World Scenario

### Cross-Installation UIDs and GIDs

شما ممکن است از چند نصب لینوکس استفاده کنید، سهولت دسترسی بر روی یک کامپیوتر یا نصب شده بر روی چند کامپیوتر. اگر از این کار استفاده کنید و فایل‌ها را از یک نصب به نصب دیگر منتقل کنید، ممکن است متوجه شوید که مالکیت فایل‌ها به طور ناگهانی تغییر کرده است. همین اتفاق ممکن است در سیستم‌های نظیر macOS نیز رخ دهد که از سیستم‌های شبیه یونیکس استفاده می‌کنند. دلیل این اتفاق این است که فایل‌سیستم‌های این سیستم‌ها از شماره‌های UID و GID برای ذخیره اطلاعات مالکیت و گروه استفاده می‌کنند و یک کاربر یا گروه می‌تواند در کامپیوترهای مختلف شناسه‌های UID یا GID متفاوتی داشته باشد، حتی اگر نام کاربری یا گروه یکسان باشد.

این مشکل احتمالاً زمانی رخ می‌دهد که از فایل‌سیستم‌های اصلی لینوکس یا یونیکس برای انتقال داده‌ها استفاده می‌شود، از جمله فایل‌سیستم‌های مبتنی بر دیسک (مانند 4ext در لینوکس یا HFS+ در macOS) یا سیستم فایل شبکه (NFS) برای دسترسی از راه دور به فایل‌ها. این مشکل احتمالاً کمتر در فایل‌سیستم‌های غیر لینوکس/یونیکس مانند FAT یا NTFS برای دیسک‌ها، یا SMB/CIFS (توسط Samba در لینوکس) برای دسترسی شبکه، رخ می‌دهد چون این فایل‌سیستم‌ها بر اساس مطابقت نام کاربری/گروه عمل می‌کنند و به شناسه‌های UID/GID وابسته نیستند.

اگر با این مشکل رویرو شدید، چندین راه حل برای حل آن وجود دارد، اما بسیاری از آن‌ها خارج از دامنه این کتاب هستند. یکی از راه حل‌ها این است که شناسه‌های UID یا GID را در یک یا چند نصب به گونه‌ای تغییر دهید که همگی یکسان باشند. فصل 13 درباره تغییر شماره UID یک کاربر با استفاده از دستور usermod و تغییر شماره GID یک گروه با استفاده از دستور groupmod صحبت می‌کند. هنگام انتقال داده‌ها از طریق دیسک‌های جابجاپذیر، استفاده از FAT یا NTFS می‌تواند راه حلی ساده باشد، البته اگر نیازی به حفظ مجوزهای Unix-style بر روی فایل‌ها نداشته باشید.

## Setting Permissions

فرآگیری امکانات مالکیت فایل بدون یک راهی برای تعیین کارهای خاص که کاربران می‌توانند با فایل‌های خود یا فایل‌های دیگران انجام دهند، بی‌معنی است. در اینجا وارد نقشهای مجوز می‌شویم. ساختار مجوزهای لینوکس بر اساس مدل یونیکس است و نیازمند توضیحی کوچک است قبل از آنکه به مسئله بپردازید. بعد از درک مبانی، می‌توانید شروع به تغییر مجوزها کنید، با استفاده از یک مدیر فایل GUI یا یک خودکار shell متنی. همچنین می‌توانید مجوزهای پیش‌فرض را برای فایل‌های جدیدی که ایجاد می‌کنید، تنظیم کنید.

## Understanding Permissions

برای درک مجوزهای Unix (و بنابراین Linux)، می‌توانید با استفاده از دستور `ls`، که فایل‌ها را در یک دایرکتوری لیست می‌کند، همراه با گزینه `-l` که یک لیست طولانی از دایرکتوری ایجاد می‌کند و شامل مجوزهای فایل‌ها است، شروع کنید. به عنوان مثال، برای دیدن یک لیست طولانی از فایل به نام `test`، می‌توانید این دستور را وارد کنید:

نکته: فصل 5، "Getting to Know the Command Line"، دستور `ls` را معرفی کرد و گزینه‌های اضافی `ls` را توضیح داد.

```
$ ls -l test
-rwxr-xr-x 1 rich users          111 Oct 13 13:48 test
```

این خط شامل چند بخش است که اطلاعات مختلفی راجع به فایل فراهم می‌کند:

ستون اول (`rwxr-xr-x` در این مثال) دسترسی‌های فایل را نشان می‌دهد. **Permissions**

ستون بعدی (1 در این مثال) تعداد لینک‌های سخت به فایل را نشان می‌دهد؛ به عبارت دیگر، تعداد نام‌های منحصر به‌فردی که می‌توانند برای دسترسی به فایل استفاده شوند.

نکته: فصل 7، "Managing Files"، پیوندها را به طور دقیق‌تر توضیح می‌دهد.

ستون بعدی (`rich` در این مثال) صاحب فایل را با نام کاربری مشخص می‌کند. **Username**

نام گروه فایل (`users` در این مثال) در ستون بعدی ظاهر می‌شود. **Group Name**

اندازه فایل در این مثال بسیار کوچک است؛ 111 بایت. **File Size**

زمان (13 Oct 13:48 در این مثال) زمانی است که آخرین بار فایل تغییر یافته است. **Time Stamp**

در نهایت، `-ls` نام فایل را نشان می‌دهد - `test` در این مثال. **Filename**

رشته‌ای که این خروجی را آغاز می‌کند (`rwxr-xr-x` در این مثال) نمایش نمادین از رشته مجوزها است. شکل 14.2 نشان می‌دهد که این رشته چگونه به چهار بخش تقسیم می‌شود:

File Type Code	Owner Permissions			Group Permissions			World Permissions		
-	r	w	x	r	-	x	r	-	x
	read	write	execute	read	write	execute	read	write	execute

Figure 14.2 A symbolic representation of file permissions is broken into four parts.

کد نوع فایل اولین نویسه کد نوع فایل است، که نوع فایل را نمایندگی می‌کند، همانطور که در جدول 14.1 خلاصه شده است. این نویسه نوع گاهی اوقات از شرح حذف می‌شود زمانی که نوع فایل مهم نباشد یا زمانی که به طریق دیگری شناسایی شود.

Table 14.1 Linux file type codes

Code	Name	Meaning
-	Normal data file	این ممکن است متن، یک برنامه قابل اجرا، گرافیک، داده‌های فشرده یا هر نوع دیگری از داده باشد.
d	Directory	دایرکتوری‌های دیسک نیز فایل هستند، اما حاوی نام‌های فایل و اشاره‌گرهای به ساختارهای داده فایل‌های نام‌گذاری شده هستند.
l	Symbolic link	لینک نمادین حاوی نام یک فایل یا دایرکتوری دیگر است. زمانی که لینوکس به لینک نمادین دسترسی پیدا می‌کند، سعی می‌کند فایل متصل به آن را بخواند.
p	Named pipe	یک پایپ امکان برقراری ارتباط دو طرفه بین دو برنامه در حال اجرای لینوکس را به صورت یک طرفه فراهم می‌کند.
s	Socket	یک سوکت مشابه یک پایپ نام‌دار است، اما اجازه ارتباط شبکه‌ای و دوطرفه را می‌دهد.
b	Block device	یک فایل دستگاه بلاک منتظر با یک دستگاه سخت‌افزار است که داده‌ها به صورت بلوک‌های بیش از یک بایت انتقال می‌یابند. دستگاه‌های دیسک (هارد دیسک‌ها، فلپی‌ها، CD-ROM‌ها و غیره) از جمله دستگاه‌های بلوک متداول هستند.
c	Character device	یک فایل دستگاه کاراکتر منتظر با یک دستگاه سخت‌افزار است که داده‌ها به صورت واحدهای یک بایت انتقال می‌یابند. به عنوان مثال، دستگاه‌های پورت

موازی و سریال RS-232 جزو این دستگاه‌های کاراکتر هستند.

نکته: بیشتر فایل‌هایی که شما احتمالاً با آنها سر و کار دارید، فایل‌های عادی، دایرکتوری‌ها و لینک‌های نمادین هستند.

در هر یک از سه مجموعه‌ی مجوز زیر، رشته‌ای وجود دارد که حضور یا عدم حضور هر یک از سه نوع دسترسی (خواندن، نوشتن و اجرا) را نشان می‌دهد:

**Owner Permissions**: این مجوزها مشخص می‌کنند که صاحب فایل چه عملیاتی را می‌تواند با فایل انجام دهد.

**Group Permissions**: این مجوزها مشخص می‌کنند که اعضای گروه فایل (که صاحب فایل نیستند) چه عملیاتی را می‌توانند با فایل انجام دهند.

**Other Permissions** یا **World Permissions**: این مجوزها مشخص می‌کنند که کاربرانی که صاحب فایل نیستند و عضو گروه آن هم نیستند، چه عملیاتی را می‌توانند با فایل انجام دهند.

در صورت حضور مجوز اجرا (execute)، این به معنای اجازه‌ی اجرای فایل به عنوان برنامه است. عدم حضور این مجوز با استفاده از علامت خط تیره (-) در رشته‌ی مجوز نمایش داده می‌شود. حضور این مجوز با حروف r برای خواندن، w برای نوشتن و x برای اجرا نشان داده می‌شود.

نکته: قرار دادن بیت اجرا (bit execute) بر روی یک فایل غیر برنامه، البته به هیچ عنوان آن فایل را به یک برنامه تبدیل نمی‌کند؛ فقط نشان می‌دهد که یک کاربر ممکن است یک فایل که یک برنامه است را اجرا کند.

دسترسی -wxr-xr-x به این معناست که فایل یک فایل داده عادی است و صاحب فایل، اعضای گروه فایل و همه کاربران دیگر می‌توانند فایل را بخوانند و اجرا کنند. تنها صاحب فایل دارای دسترسی نوشتن به فایل است.

نمایش دیگری از دسترسی‌ها نیز وجود دارد که فشرده است اما کمی گیج‌کننده است؛ این نمایش هر یک از سه گروه دسترسی رشته‌ی دسترسی (با حذف کد نوع فایل) را گرفته و آن را به یک عدد از 0 تا 7 (به صورت عدد هشتگانه یا اکتال) تبدیل می‌کند. نتیجه یک عدد اکتال سه رقمی است. هر عدد با شروع از مقدار 0 ساخته می‌شود و سپس:

- 4 اضافه می‌شود اگر دسترسی خواندن موجود باشد.
- 2 اضافه می‌شود اگر دسترسی نوشتن موجود باشد.
- 1 اضافه می‌شود اگر دسترسی اجرا موجود باشد.

کد سه رقمی حاصل، دسترسی‌ها برای صاحب، گروه و دنیا را نماینده می‌شود. جدول 14.2 نمونه‌هایی از دسترسی‌های معمول و معانی آنها را نشان می‌دهد.

نکته: این روش‌ها عملیات اعداد دودویی و منطقی را شامل می‌شوند، نه عملیات حسابی. اما توضیح حسابی آنها راحت‌تر قابل درک است.

Table 14.2 Example permissions and their interpretations

Permission string	Octal code	Meaning
rwxrwxrwx	777	دسترسی‌های خواندن، نوشتن و اجرا برای همه کاربران
rwxr-xr-X	755	دسترسی خواندن و اجرا برای همه کاربران. همچنین صاحب فایل دسترسی نوشتندار نیز دارد.
rwxr-X---	780	دسترسی خواندن و اجرا برای صاحب و گروه فایل وجود دارد. همچنین، صاحب فایل دسترسی نوشتندار نیز دارد. کاربران دیگر دسترسی به فایل ندارند.
rWX-----	700	دسترسی خواندن، نوشتندار فقط برای صاحب فایل وجود دارد؛ سایر کاربران دسترسی به فایل ندارند.
rw-rw-rw-	666	دسترسی خواندن و نوشتندار برای تمام کاربران وجود دارد، اما هیچ کدام از آنها دسترسی اجرا به فایل ندارند.
rw-rw-r--	664	دسترسی خواندن و نوشتندار برای صاحب و گروه فایل وجود دارد. تنها دسترسی خواندن برای سایر کاربران ممکن است.
rw-rw----	660	دسترسی خواندن و نوشتندار برای صاحب و گروه فایل وجود دارد. هیچ دسترسی برای سایر کاربران نیست.
rw-r--r--	644	دسترسی خواندن و نوشتندار برای صاحب فایل وجود دارد. تنها دسترسی خواندن برای سایر کاربران ممکن است.
rw-r-----	640	دسترسی خواندن و نوشتندار برای صاحب فایل وجود دارد، و دسترسی فقط خواندن برای گروه وجود دارد. هیچ دسترسی‌ای برای دیگران وجود ندارد.
rw-----	600	دسترسی خواندن و نوشتندار برای صاحب فایل وجود دارد. هیچ دسترسی‌ای برای دیگران وجود ندارد.
r-----	400	دسترسی خواندن برای صاحب فایل وجود دارد. هیچ دسترسی‌ای برای دیگران وجود ندارد.

نکته: تعداد ترکیب‌های ممکن برای دسترسی‌ها 512 است، بنابراین جدول 14.2 ناقص است. این جدول نمایش‌دهنده‌ی بیشترین و پرکاربردترین ترکیب‌های دسترسی است.

چندین مورد خاص برای دسترسی‌ها وجود دارد:

**Directory Execute Bits**: دایرکتوری‌ها از بیت اجرا استفاده می‌کنند تا اجازه ورود به دایرکتوری و دسترسی به فایل‌ها را بدهند. حتی اگر شما دسترسی خواندن فایل را داشته باشید، باید دسترسی اجرا به دایرکتوری را داشته باشید تا به فایل دسترسی پیدا کنید. این ویژگی بسیار مفید برای دایرکتوری‌هاست، بنابراین شما تقریباً همیشه بیت اجرا را در هنگام تنظیم بیت خواندن مشاهده خواهید کرد.

**Directory Write Permissions**: دایرکتوری‌ها به عنوان فایل‌هایی با استفاده از روش‌های خاص تفسیر می‌شوند. بنابراین، اگر یک کاربر به یک دایرکتوری دسترسی به نوشتن داشته باشد، او می‌تواند فایل‌ها را ایجاد، حذف یا تغییر نام دهد، حتی اگر صاحب این فایل‌ها نباشد و دسترسی به نوشتن به این فایل‌ها را نداشته باشد.

نکته: قوانین معمول برای نوشتن در دایرکتوری‌ها می‌توانند با استفاده از Sticky Bit تغییر یابند، که بعداً در بخش "Using Sticky Bits" توضیح داده می‌شود.

**Symbolic Links**: دسترسی‌های بر روی لینک‌های نمادین همیشه 777 (rwxrwxrwx) یا 777 (rwxrwxrwx) برای شامل کردن کد نوع فایل است. این دسترسی فقط بر روی فایل لینک خود اعمال می‌شود و نه بر روی فایلی که به آن لینک شده است. به عبارت دیگر، تمام کاربران می‌توانند محتويات لینک را بخوانند تا نام فایلی که به آن اشاره دارد را بفهمند، اما دسترسی‌ها بر روی فایل متصل شده توسط دسترسی‌های آن تعیین می‌شود. تغییر دسترسی‌ها بر روی یک لینک نمادین تأثیری بر روی فایل متصل شده دارد.

برای کاربر روت، بسیاری از قوانین دسترسی معمولی نقص‌هایی دارند. ابرکاربر (root) می‌تواند هر فایلی را در رایانه بخواند یا بنویسد، حتی فایل‌هایی که دسترسی آن‌ها به هیچ‌کس (یعنی فایل‌هایی که دارای دسترسی 000 هستند) را ممنوع می‌کنند. با این حال، برای اجرای فایل‌های برنامه نیاز به تنظیم بیت اجرا همچنان وجود دارد.

## Setting Permissions in a File Manager

فرایند تنظیم دسترسی‌ها در یک مدیر فایل شبیه به فرایند تنظیم مالکیت یک فایل است:

- معمولاً شما این تنظیمات را با استفاده از همان دیالوگ که برای تنظیم مالکیت استفاده می‌شود، مانند دیالوگ GNOME Files که در شکل 14.1 نشان داده شده است، تنظیم می‌کنید.
- شما نیازی به حق دسترسی روت برای تنظیم دسترسی‌های فایل‌هایی که مالک آن‌ها هستید ندارید.
- برای این کار فقط بر روی فایل‌هایی که مالک آن‌ها نیستید باید از دسترسی روت استفاده کنید.

همانطور که در شکل 14.1 قبلًا دیده شد، سه گزینه دسترسی وجود دارد که به مالک، گروه و دیگران مربوط می‌شود:

- گزینه مالک دو گزینه دارد: فقط خواندن و خواندن و نوشتن.
- گزینه‌های گروه و دیگران هر دو گزینه‌های فقط خواندن و خواندن و نوشتن به علاوه گزینه بدون دسترسی را فراهم می‌کنند. شما می‌توانید از این گزینه‌ها برای تنظیم بیت‌های دسترسی خواندن و نوشتن بر روی فایل خود استفاده کنید.

GNOME Files نیاز به تنظیم execute bit را به صورت جداگانه دارد که با علامت زدن جعبه Allow Executing File as Program انجام می‌دهد. این جعبه‌ای که تمامی سه بیت دسترسی اجرا را تنظیم می‌کند؛ شما نمی‌توانید با GNOME Files دقیق‌تر دسترسی اجرا را کنترل کنید. همچنین شما نمی‌توانید دسترسی‌های اجرا را بر روی دایرکتوری‌ها با GNOME Files تنظیم کنید.

نکته: جزئیات در تنظیم دسترسی‌ها در مدیرهای فایل دیگر ممکن است متفاوت باشد، اما اصول آنها مشابه اصول توضیح داده شده برای GNOME Files هستند.

## Setting Permissions in a Shell

در یک پوسته حالت متنی، می‌توانید از دستور chmod برای تغییر دسترسی‌ها استفاده کنید. این دستور نسبتاً پیچیده است، اصلی‌ترین دلیل آن هم پیچیدگی روش‌های مختلفی است که می‌توانید دسترسی‌ها را تغییر دهید. شما می‌توانید دسترسی‌ها را به دو شکل مشخص کنید: به عنوان یک عدد اوکتال یا به صورت نمادین، که یک مجموعه کد مرتبط با نمایش رشته‌ای از دسترسی‌ها است.

نکته: دستور chmod به معنای تغییر حالت است، که حالت به معنای دیگری برای دسترسی‌ها است.

نمایش داده شده در اینجا یک نحوه تغییر سطوح دسترسی به صورت عددی است که به آن اشاره کردیم و در جدول 14.2 خلاصه شده است. به عنوان مثال، برای تغییر سطوح دسترسی در فایل report.txt به `-rwxr--r-` می‌توانید دستور زیر را وارد کنید:

```
$ chmod 644 report.txt
```

در مقابل، لینک نمادین از سه بخش تشکیل شده است:

- یک کد که دسترسی مورد نظر را نشان می‌دهد؛ u برای کاربر (یعنی صاحب فایل)، g برای گروه، o برای کاربران دیگر، و a برای تمامی دسترسی‌ها
- نمادی که نشان می‌دهد که آیا می‌خواهید دسترسی را اضافه کنید (+)، حذف کنید (-)، یا حالت را برابر با (=) مقدار مشخص شده قرار دهید
- یک کد که دسترسی مورد نظر را مشخص می‌کند، مانند نمادهای متداول w، r، یا x، یا نمادهای دیگر برای عملیات‌های پیچیده‌تر

استفاده از حالت نمادین با دستور chmod ممکن است گیج‌کننده باشد، بنابراین ما اینجا به طور کامل توضیح نمی‌دهیم؛ با این حال، شما باید با چند نوع متداول استفاده آشنا باشید، که در جدول 14.3 خلاصه شده است. حالت‌های نمادین از حالت‌های اعدادی قابل تغییرتر هستند زیرا می‌توانید حالت‌های نمادین را مشخص کنید که دسترسی‌های موجود را تغییر دهند، مانند اضافه کردن یا حذف دسترسی اجرا بدون اثر گذاشتن بر دیگر دسترسی‌ها. همچنین می‌توانید فقط دسترسی‌های کاربر، گروه یا دنیا را بدون اثر روی دیگران تنظیم کنید. در حالت‌های اعدادی، شما باید همه سه بیت دسترسی را به یک مقدار مشخص تنظیم کنید.

نکته: مانند دستورات chown و chgrp، شما می‌توانید از گزینه R-(یا recursive) در دستور chmod استفاده کنید تا این دستور را بر روی یک درخت کامل از دایرکتوری‌ها اجرا کنید.

Table 14.3 Examples of symbolic permissions with chmod

Command	Initial permissions	End permissions
chmod a+x bigprogram	rwxr--r--	rwxr-xr-x
chmod ug=rw report.txt	r-----	rW-rW----
chmod o-rwx bigprogram	rwxrwxr-x	rwxrwx---
chmod g-w,o-rw report.txt	rwxr--r--	rW-r-----

## Setting the umask

The umask، یا user mask، تعیین‌کننده‌ی مجوزهای پیش‌فرض برای فایل‌ها و دایرکتوری‌های جدید است. این مقداری است که از مجوزهای 666 (rw-rw-rw-) برای ایجاد فایل‌های جدید یا از 777 (rwxrwxrwx) برای ایجاد دایرکتوری‌های جدید کم می‌شود. به عنوان مثال، اگر umask 022 باشد، فایل‌ها به طور پیش‌فرض با مجوز 644 ایجاد می‌شوند و دایرکتوری‌های جدید با مجوز 755 خواهند بود. توجه کنید که عمل حذف یک منطقی‌ترین عملگر، به معنای کمی از یک اعداد 7 است که به مجوزهای متناظر rwx اعمال می‌شود، اما برای فایل‌ها که مقدار اولیه آنها به rw- است، نه 1- (که بی‌معنی است).

شما می‌توانید با استفاده از دستور umask را تنظیم کنید، مانند umask 022. به طور معمول، این دستور در یک فایل پیکربندی سیستم، مانند /etc/profile، یا در یک فایل پیکربندی کاربر، مانند ~/.bashrc ظاهر می‌شود.

## Using Special Permission Bits and File Features

وقتی که درخت دایرکتوری لینوکس را بررسی می‌کنید، با برخی از انواع فایل‌هایی روبرو می‌شوید که نیاز به توجه ویژه دارند. بعضی اوقات ممکن است شما فقط می‌خواهید آگاه شوید که چگونه این فایل‌ها را باید بررسی کنید، زیرا اینها از آنچه که انتظار می‌رود در مطلب فصل 8 مطرح شده، انحراف دارند. در موارد دیگر، شما ممکن است نیاز داشته باشید که چگونگی استفاده از دستور ls یا دیگر دستورها را برای مقابله با این فایل‌ها و دایرکتوری‌ها تنظیم کنید. این موارد ویژه شامل "sticky bit"، پنهان کردن فایل‌ها از دید، دریافت لیست‌های طولانی از دایرکتوری‌ها و استفاده از مجوزهای خاص اجرا می‌شود.

## Using Sticky Bits

قبل از ورود به توضیح sticky bits، بهتر است با توضیح اینکه چرا آن‌ها نیاز داریم، شروع کنیم. در نظر بگیرید دستورات زیر را که در یک سیستم با چند فایل و زیرشاخه به یک شکل خاص اجرا می‌شود:

```
$ whoami  
rich  
  
$ ls -l  
  
total 0  
  
drwxrwxrwx 2 root root 80 Oct 14 17:58 subdir  
  
$ ls -l subdir/  
  
total 2350  
  
-rw-r----- 1 root root 2404268 Oct 14 17:59 report.txt
```

این دستورات پیکربندی فعلی را تعیین می‌کنند؛ شناسه کاربری فعلی rich است و دایرکتوری فعلی یک زیرشاخه به نام `subdir` دارد که توسط root مالکیت دارد، اما rich و همه کاربران سیستم دسترسی کامل به آن دارند. این زیردایرکتوری یک فایل به نام `report.txt` دارد که توسط root مالکیت دارد و به آن دسترسی ندارد. می‌توانید با استفاده از دستور `touch` بررسی کنید که rich دسترسی نوشتن به این فایل ندارد:

```
$ touch subdir/report.txt  
  
touch: cannot touch `subdir/report.txt': Permission denied
```

این پیام خطا نشان می‌دهد که rich نمی‌توانست به `subdir/report.txt` بنویسد. به نظر می‌رسد که فایل از دست دستکاری امن است. اما اینطور نیست! این را امتحان کنید:

```
$ rm subdir/report.txt  
  
$ ls -l subdir/  
  
total 0
```

دستور `rm` پیام خطا نداد و بررسی بعدی از `subdir` نشان می‌دهد که اکنون خالی است - به عبارت دیگر، rich توانست فایل را حذف کند حتی بدون دسترسی نوشتن به آن! این ممکن است به نظر برسد که یک باگ است - در نهایت، اگر نتوانید به یک فایل بنویسید، فکر می‌کنید نباید بتوانید آن را حذف کنید. با این حال به یاد داشته باشید که دایرکتوری‌ها فقط یک نوع ویژه از فایل‌ها هستند که نام فایل‌های دیگر را در خود نگه

می‌دارند و به ساختارهای داده پایین‌تر آنها اشاره می‌کنند. بنابراین، اصلاح کردن یک فایل نیازمند دسترسی نوشتن به فایل است، اما ایجاد یا حذف یک فایل نیازمند دسترسی نوشتن به دایرکتوری است که در آن قرار دارد. در این مثال، rich دسترسی نوشتن به دایرکتوری subdir را دارد اما دسترسی نوشتن به فایل report.txt درون آن دایرکتوری ندارد. بنابراین، rich می‌تواند فایل را حذف کند اما نمی‌تواند آن را اصلاح کند. این نتیجه یک باگ نیست؛ بلکه یک ویژگی غیرمعمول است.

اگرچه فایل سیستم‌های Linux برای کار با این شیوه طراحی شده‌اند، اما چنین رفتاری همیشه مطلوب نیست. راه برای ایجاد نتیجه‌ای مفهومی‌تر استفاده از sticky bit است که یک اجازه ویژه است که این رفتار را تغییر می‌دهد. با تنظیم sticky bit با دستور chown، به یکی از دو روش زیر:

**Using an Octal Code**: با پیشوند یک عدد دیگر می‌توانید هر یک از سه بیت اجازه ویژه را تنظیم کنید، یکی از آنها sticky bit است. کد برای sticky bit، یک است، بنابراین می‌توانید از یک کد هشتگانه که با ۱ شروع می‌شود، مانند ۱۷۵۵، برای تنظیم sticky bit استفاده کنید. تعیین یک مقدار ۰، مانند ۰۷۵۵ sticky bit را حذف می‌کند.

نکته: اعداد فرد دیگر نیز sticky bit را تنظیم خواهند کرد، اما همچنین بیت‌های اجازه ویژه اضافی را نیز تنظیم خواهند کرد که به زودی در "Using Special Execute Permissions" توضیح داده خواهد شد.

**Using a Symbolic Code**: برای تنظیم sticky bit به دسترسی‌های جهانی، از کد نمادین t استفاده کنید، مانند chmod o+t subdir. می‌توانید با استفاده از علامت منفی نیز sticky bit را حذف کنید، مانند -t subdir

با بازنشانی فایل و اعطای sticky bit، می‌توانید اثر آن را ببینید:

```
$ ls -l  
total 0  
drwxrwxrwt 2 root root 80 Oct 14 18:25 subdir
```

```
$ ls -l subdir/  
total 304  
-rw-r--r-- 1 root root 2404268 Oct 14 18:25 report.txt
```

```
$ rm subdir/report.txt
```

```
rm: cannot remove `subdir/report.txt': Operation not permitted
```

در این مثال، با وجود دسترسی کامل خواندن/نوشتن `subdir` به `rich`، نمی‌تواند فایل‌های کاربر دیگری را در آن دایرکتوری حذف کند.

می‌توانید یک دایرکتوری با sticky bit تنظیم شده را با تغییر کوچکی در حالت نمادین نمایش داده شده توسط `ls` شناسایی کنید. `world execute bit` به جای `X` به صورت `t` نشان داده می‌شود. در این مثال، نتیجه آن است که دسترسی `subdir` به صورت `drwxrwxrwt` نمایش داده می‌شود به جای `WX`.

sticky bit به ویژه برای دایرکتوری‌هایی که توسط بسیاری از کاربران به اشتراک گذاشته می‌شود، اهمیت دارد. به عنوان مثال، این یک ویژگی استاندارد در `/tmp` و `/var/tmp` است، زیرا بسیاری از کاربران فایل‌های موقت را در این دایرکتوری‌ها ذخیره می‌کنند و نمی‌خواهید یک کاربر بتواند فایل‌های موقت کاربر دیگری را حذف کند. اگر می‌خواهید که کاربرانی که در یک پروژه همکاری می‌کنند بتوانند فایل‌ها را در دایرکتوری‌های خانه‌ای هم‌مدیگر نوشته و خوانند، در نظر داشته باشید که sticky bit را در این دایرکتوری‌های خانه یا زیر‌دایرکتوری‌هایی که کاربران فایل‌ها را به اشتراک گذاشته‌اند، تنظیم کنید.

نکته: اگر شما `/var/tmp` یا `/tmp` را حذف کرده و نیاز به بازسازی آن دارید، حتماً از قرار دادن sticky bit بر روی دایرکتوری جایگزین جدیدتان اطمینان حاصل کنید!

## Using Special Execute Permissions

برنامه‌ها در لینوکس به عنوان فایل‌های قابل اجرا شناخته می‌شوند. این فایل‌ها به کمک اعتبارات کاربری شما اجرا می‌شوند که به طور کلی چیز مفیدی است؛ ارتباط پروسه‌های در حال اجرا با کاربران خاص بخشی از مدل امنیتی لینوکس است. با این حال، گاهی برنامه‌ها نیاز به اجرا با امتیازات بالا دارند. به عنوان مثال، برنامه `passwd` که برای تنظیم رمز عبور کاربران استفاده می‌شود، برای نوشتن و گاهی برای خواندن فایل‌های پیکربندی نیاز به اجرا به عنوان `root` دارد. بنابراین، حتی زمانی که کاربران عادی این برنامه را برای تغییر رمز خود اجرا می‌کنند، باید دسترسی `root` را داشته باشد تا به درستی کار کند.

برای انجام این کار، دو بیت اختصاص داده شده ویژه وجود دارد، مشابه به بیت چسبندگی که قبلًاً توضیح داده شد:

**(SUID) یا Set User ID گزینه**: (SUID) یا set user ID به لینوکس می‌گوید که برنامه را با دسترسی‌های کسی که فایل را مالک آن است، اجرا کند به جای اجرای برنامه با دسترسی‌های کاربری که برنامه را اجرا کرده است. به عنوان مثال، اگر یک فایل توسط `root` مالکیت داشته باشد و بیت SUID آن تنظیم شود، برنامه با دسترسی‌های `root` اجرا می‌شود و بنابراین می‌تواند به هر فایلی روی کامپیوتر دسترسی داشته باشد. برخی از سرورها و برنامه‌های سیستم دیگر به این شیوه اجرا می‌شوند، که اغلب به آن SUID `root` می‌گویند. برنامه‌های SUID با `S` در موقعیت بیت اجرا کننده مالک در رشته دسترسی نمایش داده می‌شوند، مانند `rwsr-xr-x`.

حال اجرا را به گروه فایل تنظیم می‌کند. این گزینه با `s` در موقعیت بیت اجرا کننده گروه در رشته دسترسی نمایش داده می‌شود، مانند `rwxr-sr-x``. وقتی روی یک دایرکتوری تنظیم شود، گزینه SUID مطمئن می‌شود که همه فایل‌هایی که در آن دایرکتوری ایجاد می‌شوند، به گروه دایرکتوری خود تعلق می‌گیرند، به جای گروه فردی که فایل را ایجاد کرده است.

می‌توانید این بیت‌ها را با استفاده از دستور `chmod` تنظیم کنید:

**Using an Octal Code:** با استفاده از اعداد اول هشتگانه چهار رقمی، مقدار اول را به 4 تنظیم کنید تا بیت SUID را تنظیم کنید، به 2 تا بیت SUID را تنظیم کنید، یا به 6 تا هر دو بیت را تنظیم کنید. به عنوان مثال، 4755 بیت SUID را بر روی یک فایل اجرایی تنظیم می‌کند، اما بیت SUID را نه.

**Using a Symbolic Code:** از کد نمادین `s`، به همراه `u` برای مشخص کردن بیت SUID، و `g` برای مشخص کردن بیت SUID، یا هر دو برای تنظیم هر دو بیت استفاده کنید. به عنوان مثال، تایپ کردن `chmod u+s myprog` بیت SUID را بر روی `myprog` تنظیم می‌کند، در حالی که `chmod ug-s myprog` هر دو بیت SUID را حذف می‌کند.

معمولًاً نیازی به تنظیم یا حذف این بیت‌ها ندارید؛ برنامه مدیریت بسته این بیت‌ها را به درستی تنظیم می‌کند هنگام نصب یا به روزرسانی یک برنامه. ممکن است نیاز به تغییر این بیت‌ها داشته باشد اگر به اشتباه این بیت‌ها بر روی فایل‌ها تنظیم شده یا حذف شده باشد. در برخی موارد ممکن است بخواهید یا نیاز داشته باشد که این مقادیر را در فایل‌های برنامه که از کد منبع کامپایل شده‌اند تنظیم کنید یا اگر نیاز به اصلاح رفتار یک برنامه دارید. با این حال بسیار موازن باشید. اگر بیت SUID یا SUID را بر روی یک برنامه عادی تنظیم کنید، برنامه با امتیازات بیشتری اجرا خواهد شد. اگر برنامه دارای باگ باشد، این باگ‌ها قادر خواهند بود بیشترین آسیب را بزنند. اگر به طور تصادفی این مجوزها را حذف کنید، نتایج میتواند همانقدر بد باشد. برنامه‌هایی مانند `sudo`، `passwd` و `su` همه بر اساس بیت SUID آن‌ها عمل می‌کنند، بنابراین حذف این ویژگی ممکن است باعث متوقف شدن کار آن‌ها شود.

## Hiding Files from View

اگر شما از ویندوز استفاده می‌کنید، ممکن است با مفهوم بیت پنهان آشنا باشید که فایل‌ها را از دید مدیرهای فایل، دستور DIR و بیشتر برنامه‌ها پنهان می‌کند. اگر به دنبال مفهوم مشابهی در لینوکس هستید، حداقل به صورت یک ویژگی اختصاصی فایل سیستم آن را نمی‌یابید. به جای آن، لینوکس از یک کنوانسیون نام‌گذاری فایل برای پنهان کردن فایل‌ها استفاده می‌کند؛ اکثر ابزارها، مانند `ls`، فایل‌ها و دایرکتوری‌ها را از دید پنهان می‌کنند اگر نام آن‌ها با نقطه (.) شروع شود. بنابراین، `ls` فایل `afile.txt` را نمایش می‌دهد اما `afile.txt` را نمی‌آورد.

بیشتر فایل منیجرها و دیالوگ‌هایی که با فایل‌ها سر و کار دارند، نیز این نوع فایل‌های نقطه‌دار را پنهان می‌کنند، هرچند که این عمل عمومی نیست.

بسیاری از برنامه‌های کاربردی از این ویژگی استفاده می‌کنند تا فایل‌های پیکربندی خود را از پراکندگی در نمایشگر شما جلوگیری کنند. به عنوان مثال، `~/.bashrc` فایل پیکربندی کاربر Bash است، فایل‌های `~/.evolution` در دایرکتوری `~/evolution` قرار دارند و `~/.fonts.conf` اطلاعات پیکربندی فونت خاص کاربر را نگه می‌دارد.

می‌توانید به روش‌های مختلفی فایل‌های نقطه‌دار را مشاهده کنید، به توجه به برنامه مورد نظر. برخی ابزارهای GUI دارای یک جعبه انتخابی در تنظیمات پیکربندی‌شان هستند که می‌توانید آن را فعال کنید تا برنامه نمایش گذاردن این فایل‌ها مجبور شود. در خط فرمان، می‌توانید از گزینه `-a` در `ls` استفاده کنید:

```
$ ls -l  
total 0  
drwxrwxrwt  2  root  root      80 Dec 14 18:25 subdir  
  
$ ls -la  
total 305  
drwxr-xr-x  3  kirk  users     104 Dec 14 18:44 .  
drwxr-xr-x  3  kirk  users     528 Dec 14 18:21 ..  
-rw-r--r--   1  kirk  users  309580 Dec 14 18:44 .report.txt  
drwxrwxrwt  2  root  root      80 Dec 14 18:25 subdir
```

این مثال فایل `report.txt` را در دایرکتوری فعلی نشان می‌دهد. همچنین دو فایل دایرکتوری پنهان نیز نشان داده می‌شود. اولین فایل، .. به دایرکتوری فعلی اشاره دارد. دومین فایل، .. به دایرکتوری والد اشاره دارد.

نکته: به یاد آورید که در فصل ششم، "Managing Hardware" .. مرجع دایرکتوری نسبی است. این ورودی پنهان دلیل کار کردنش است.

توجه کنید که تغییر نام یک فایل به نحوی که با نقطه شروع شود، آن را پنهان می‌کند، اما این کار باعث می‌شود که فایل برای هر برنامه‌ای که از نام اصلی استفاده می‌کند غیر قابل دسترس شود. به عبارت دیگر، اگر فایل `report.txt` را به `report.txt` تغییر نام دهید و اگر برنامه‌ای دیگر به فایل به عنوان `report.txt` ارجاع دهد، این ارجاع دیگر کار نمی‌کند. باید نقطه ابتدایی را در هر ارجاع به فایل پنهان در نظر بگیرید.

## Viewing Directories

فصل ششم دستور `ls` و بسیاری از گزینه‌های آن را معرفی کرد. یکی از این گزینه‌ها که در این نقطه به توضیح نیاز دارد `d`-است. اگر در یک دایرکتوری باشید که شامل بسیاری از زیردایرکتوری‌ها است و اگر از `wildcard` دستور `ls` استفاده کنید که با یک یا چندین زیردایرکتوری مطابقت دارد، ممکن است نتیجه غیرمنتظره‌ای

دربیافت کنید؛ خروجی فایل‌های موجود در زیردایرکتوری‌های مطابق نمایش داده می‌شود، به جای اطلاعات خود زیردایرکتوری‌ها - به عنوان مثال، اگر در یک دایرکتوری با دو زیردایرکتوری، 1subdir و 2subdir، شروع کنید:

```
$ ls -l subdir*
```

subdir1:

total 304

```
-rw-r--r-- 1 kirk users 309580 Dec 14 18:54 report.txt
```

subdir2:

total 84

```
-rw-r--r-- 1 kirk users 86016 Dec 14 18:54 mypaper.doc
```

اگر به جای اطلاعات مربوط به محتوای زیردایرکتوری‌ها، اطلاعاتی در مورد خود زیردایرکتوری‌ها را می‌خواهید، می‌توانید از گزینه d- استفاده کنید:

```
$ ls -ld subdir*
```

```
drwxr-xr-x 2 kirk users 80 Dec 14 18:54 subdir1
```

```
drwxr-xr-x 2 kirk users 80 Dec 14 18:54 subdir2
```

# Chapter 15

## Managing Network Connections

این روزها تقریباً ضرورت دارد که سیستم لینوکس شما به یک نوع شبکه متصل شود. اگر این نیاز به اشتراک‌گذاری فایل‌ها و پرینترها در شبکه محلی باشد یا نیاز به اتصال به اینترنت برای دانلود به روزرسانی‌ها و پچ‌های امنیتی باشد، بیشتر سیستم‌های لینوکس دارای اتصالی به یک نوع شبکه هستند.

این فصل به بررسی نحوه پیکربندی سیستم لینوکس شما برای اتصال به یک شبکه می‌پردازد، همچنین به رفع مشکلات اتصال شبکه پرداخته می‌شود اگر موارد به سرعت پیش بروند. چندین روش مختلف برای پیکربندی تنظیمات شبکه در لینوکس وجود دارد و شما باید همه آنها را برای آزمون مهارت‌های اساسی لینوکس بدانید. ابتدا، تنظیمات ابتدایی مورد نیاز برای اتصال به شبکه را پوشش می‌دهیم. سپس، ابزارهای مختلفی که در اختیار دارید برای پیکربندی تنظیمات شبکه را بررسی می‌کنیم. در ادامه، این فصل به روش‌های ساده اشکال‌زدایی شبکه می‌پردازد که می‌توانید از آنها استفاده کنید تا اگر هر گونه مشکلی پیش بیاید، آن را پیدا کنید.

### Configuring Network Features

برای تعامل با یک شبکه در سیستم لینوکس خود، نیاز به پیکربندی پنج اطلاعات اصلی دارید:

- آدرس میزبان
- آدرس Subnet شبکه
- روتر پیش‌فرض (گاهی به آن Gateway گفته می‌شود)
- نام میزبان سیستم (host name)
- آدرس سرور DNS برای تبدیل نامهای میزبان

شما سه روش برای پیکربندی این اطلاعات در سیستم‌های لینوکس دارید:

- ویرایش دستی فایل‌های پیکربندی شبکه
- استفاده از ابزار گرافیکی که با توزیع لینوکس شما همراه است
- استفاده از ابزارهای کامند لاین

تلash برای ویرایش دستی فایل‌های پیکربندی شبکه بهتر است برای سیستم ادمین‌های پیشرفته باقی بماند و فراتر از محدوده این کتاب است. برای بیشتر کاربران معمولی لینوکس، نیازی به دستکاری فایل‌های پیکربندی نخواهید داشت؛ ابزارهای گرافیکی و خط فرمان می‌توانند تمام این کارها را برای شما انجام دهند. بخش‌های بعدی هم ابزارهای گرافیکی و هم ابزارهای خط فرمان را مرور می‌کنند.

نکته: شایان ذکر است که امروزه دو نوع طرح آدرس شبکه IP مورد استفاده قرار می‌گیرند. طرح آدرس قدیمی به صورت فنی IPv4 نامیده می‌شود اما معمولاً فقط به عنوان IP شناخته می‌شود. این طرح از 32 بیت برای

نشان دادن آدرس میزبان استفاده می‌کند. این 32 بیت معمولاً به چهار مقدار 8 بیتی تقسیم می‌شود که با مقادیر اعشاری نشان داده می‌شوند و با نقطه از هم جدا می‌شوند (مانند 192.168.1.5). به دلیل کمبود آدرس‌های 32 بیتی منحصر به فرد برای تخصیص به میزبان‌ها در اینترنت، 6IPV6 ایجاد شد. این طرح از 128 بیت برای آدرس‌ها استفاده می‌کند. این مقادیر معمولاً به صورت هشت گروه از چهار رقم هگزادسیمال نشان داده می‌شوند که با کولون از هم جدا می‌شوند (مانند 1602:1:aa10:fa10:cf2500:eeb0:e900:33ce0).

## Graphical Tools

ابزار Network Manager یک برنامه محبوب است که توسط بسیاری از توزیع‌های لینوکس برای ارائه یک رابط گرافیکی جهت تعریف اتصالات شبکه استفاده می‌شود. Network Manager به صورت خودکار در زمان بوت شدن سیستم شروع به کار می‌کند و به عنوان یک آیکون در ناحیه سینی سیستم دسکتاپ ظاهر می‌شود.

اگر سیستم شما یک اتصال شبکه سیمی را شناسایی کند، آیکون به صورت یک شبکه مینی با بلوک‌های متصل به هم ظاهر می‌شود. اگر سیستم شما یک اتصال شبکه بی‌سیم را شناسایی کند، آیکون به صورت یک سیگنال رادیویی خالی نمایش داده می‌شود. هنگامی که روی آیکون کلیک می‌کنید، یک لیست از شبکه‌های بی‌سیمی که توسط کارت شبکه شناسایی شده‌اند، ظاهر می‌شود (همانطور که در شکل 15.1 نشان داده شده است).

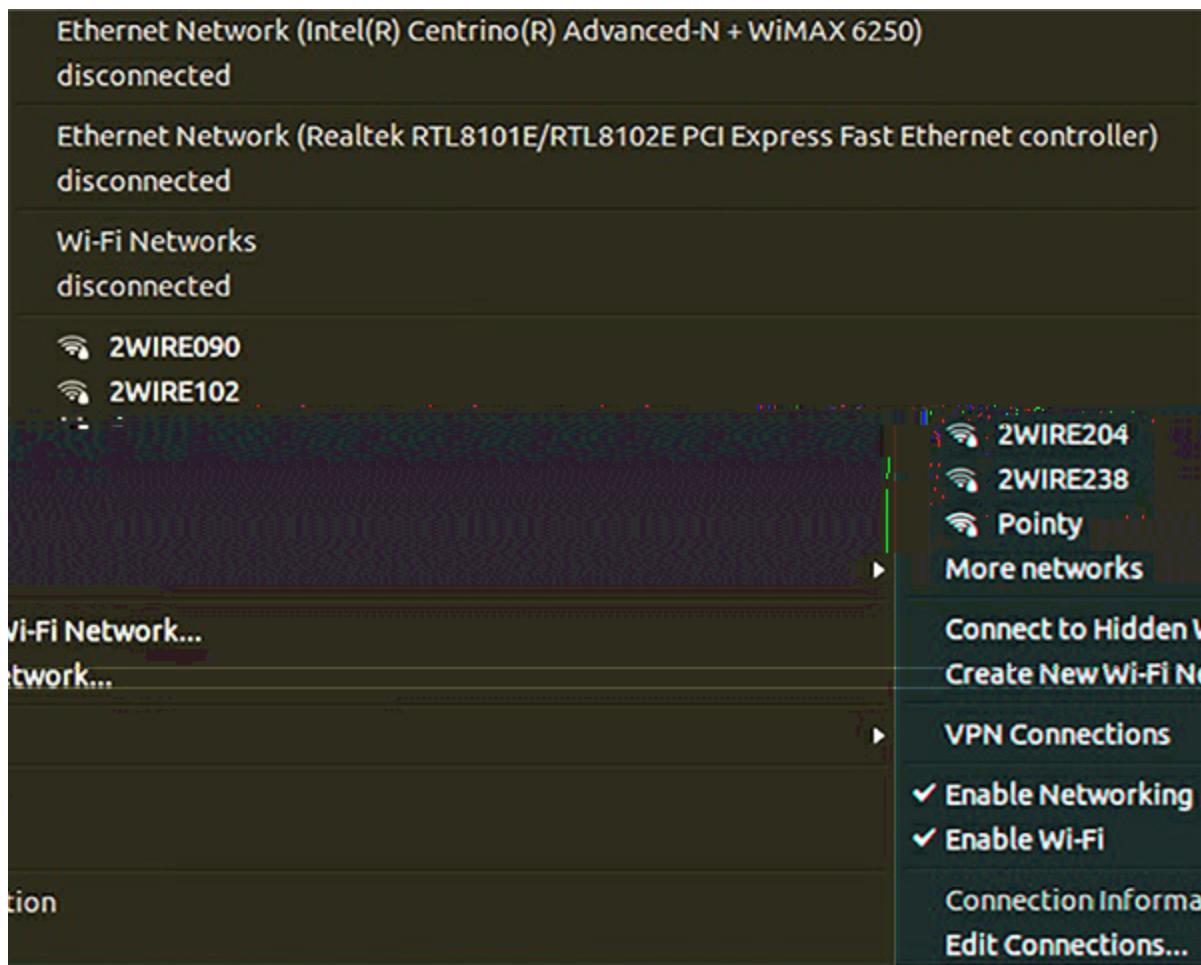


Figure 15.1: Network Manager showing a wireless network connection

روی نقطه دسترسی (access point) خود کلیک کنید تا آن را از لیست انتخاب کنید. اگر نقطه دسترسی شما رمزگذاری شده باشد، از شما خواسته می‌شود که رمز عبور را وارد کنید تا به شبکه دسترسی پیدا کنید.

بعد از اینکه سیستم شما به یک نقطه دسترسی بی‌سیم متصل شد، آیکون به صورت یک سیگنال رادیویی ظاهر می‌شود. روی آیکون کلیک کنید و سپس Edit Connections را انتخاب کنید تا تنظیمات اتصال شبکه برای سیستم را ویرایش کنید، همانطور که در شکل 15.2 نشان داده شده است.

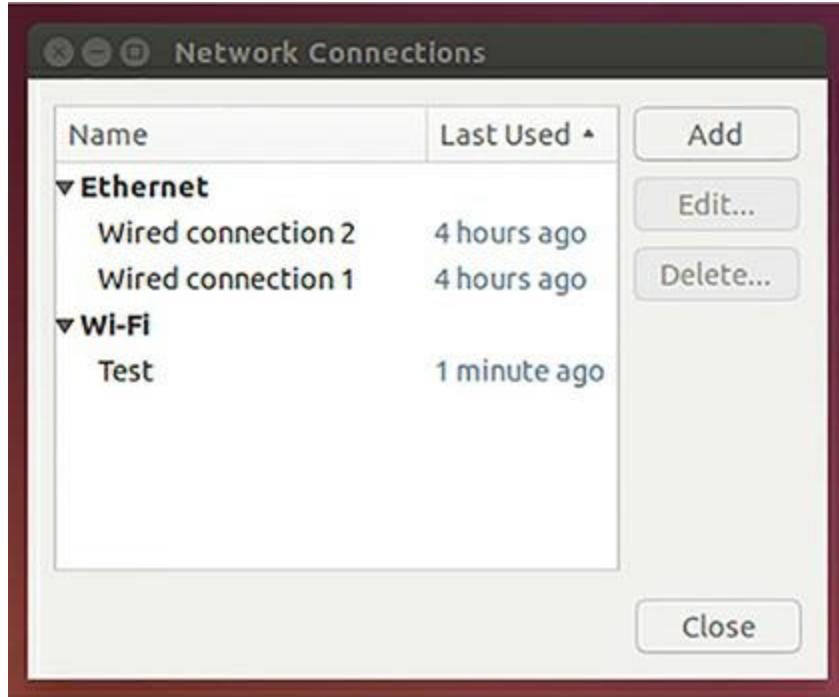


Figure 15.2: The Network Connections dialog

شبکه‌ای را که می‌خواهید پیکربندی کنید (چه بی‌سیم و چه سیمی) انتخاب کنید و سپس برای تغییر تنظیمات فعلی روی دکمه Edit کلیک کنید. ابزار Network Manager به شما اجازه می‌دهد تا با استفاده از گزینه پیکربندی دستی، آدرس میزبان، آدرس Subnet شبکه، روتر پیشفرض و نام سرورهای DNS را مشخص کنید، یا می‌توانید تنظیمات را به گونه‌ای تنظیم کنید که از پروتکل DHCP برای تعیین تنظیمات استفاده کند. Network Manager به طور خودکار فایل‌های پیکربندی شبکه مربوطه را با تنظیمات بهروز شده به روزرسانی می‌کند.

### Real World Scenario

**Manual DNS Configurations**

یکی از موضوعاتی که آزمون Linux Essentials انتظار دارد درباره فایل‌های پیکربندی شبکه بدانید، DNS است. شما می‌توانید سرور DNS را به صورت دستی تعریف کنید تا سیستم بتواند از نامهای میزبان DNS استفاده کند. خوشبختانه، این یک استاندارد است که همه سیستم‌های لینوکس از آن پیروی می‌کنند و در فایل پیکربندی /etc/resolv.conf مدیریت می‌شود:

```
domain mydomain.com
search mytest.com
nameserver 192.168.1.1
```

ورودی domain نام دامنه اختصاص داده شده به شبکه را تعریف می‌کند. به طور پیش‌فرض، سیستم این نام دامنه را به هر نام میزبانی که مشخص می‌کنید اضافه می‌کند. ورودی search هر دامنه اضافی که برای جستجوی نامهای میزبان استفاده می‌شود را تعریف می‌کند. ورودی nameserver جایی است که شما سرور DNS اختصاص داده شده به شبکه خود را مشخص می‌کنید. برخی شبکه‌ها ممکن است بیش از یک سرور DNS داشته باشند؛ فقط ورودی‌های nameserver اضافی را در فایل اضافه کنید. با این حال، مراقب باشید، زیرا ممکن است این فایل دفعه بعدی، که سیستم لینوکس شما بوت می‌شود ریست شود، بسته به اینکه توزیع لینوکس شما چگونه پیکربندی شده است. برای کمک به سرعت بخشیدن به اتصالات به میزبان‌های معمولاً استفاده شده، نامهای میزبان و آدرس‌های IP آن‌ها را به صورت دستی در فایل /etc/hosts در سیستم لینوکس خود وارد کنید. فایل /etc/nsswitch.conf تعیین می‌کند که آیا سیستم لینوکس این فایل را قبل یا بعد از استفاده از DNS برای جستجوی نام میزبان بررسی کند.

## Command-Line Tools

اگر شما با محیط کاربری گرافیکی دسکتاپ کار نمی‌کنید، نیاز دارید تا از ابزارهای کامند لاین لینوکس برای تنظیم اطلاعات پیکربندی شبکه استفاده کنید. ابزارهای مختلفی برای کامند لاین در دسترس شما هستند. این بخش ابزارهایی را پوشش می‌دهد که به احتمال زیاد با آن‌ها برخورد خواهید کرد (و آن‌هایی که احتمالاً در آزمون Linux Essentials خواهید دید).

## Network Manager Command-Line Tools

دو ابزار کامند لاین ارائه می‌دهد:

- nmtui – یک ابزار منوی متنی ساده فراهم می‌کند.
- nmcli – یک ابزار کامند لاین متنی تنها فراهم می‌کند.

هر دوی این ابزارها به شما در فرآیند تنظیم اطلاعات شبکه مورد نیاز برای سیستم لینوکستان کمک می‌کنند. ابزار nmtui یک نسخه ساده شده از ابزار گرافیکی را نمایش می‌دهد که در آن می‌توانید یک رابط شبکه انتخاب کرده و ویژگی‌های شبکه را به آن اختصاص دهید، همانطور که در شکل 15.3 نشان داده شده است.

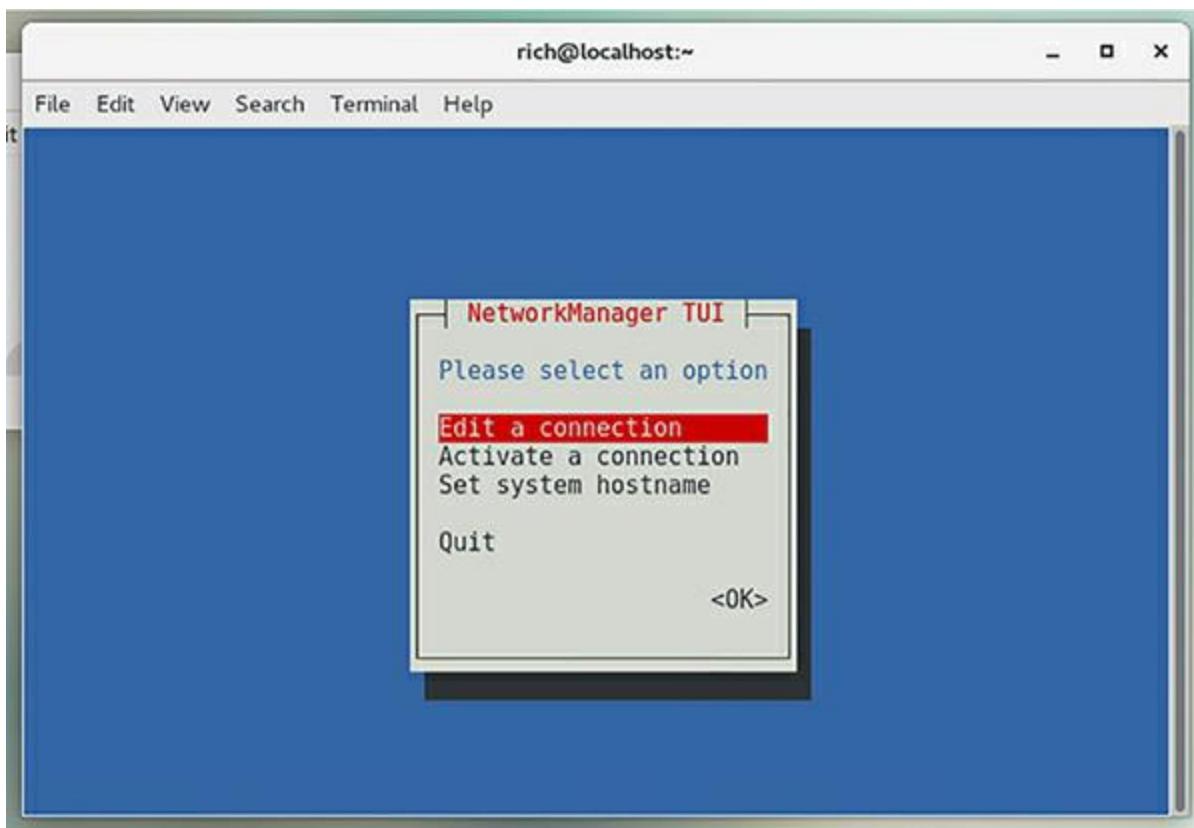


Figure 15.3: The Network Manager nmtui command-line tool

``nmcli'' تلاشی برای استفاده از قابلیت‌های گرافیکی ندارد؛ فقط یک رابط کامند لاین فراهم می‌کند که شما می‌توانید در آن تنظیمات شبکه را مشاهده و تغییر دهید. به طور پیش‌فرض، این دستور وضعیت فعلی دستگاهها و تنظیمات آن‌ها را نمایش می‌دهد، همانطور که در لیست 15.1 نشان داده شده است.

Listing 15.1: The default output of the nmcli command

```
$ nmcli  
enp0s3:      connected to enp0s3  
              "Intel 82540EM Gigabit Ethernet Controller (PRO/1000  
MT  
Desktop  
Adapter)  
          ethernet (e1000), 08:00:27:73:1C:6D, hw, mtu 1500  
          ip4 default  
          inet4 10.0.2.15/24  
          route4 0.0.0.0/0  
          route4 10.0.2.0/24  
          inet6 fe80::5432:eddb:51ea:fb44/64  
          route6 ff00::/8  
          route6 fe80::/64  
          route6 fe80::/64
```

دستور nmcli از آپشن های کامند لاین استفاده میکند تا به شما قابلیت تنظیمات شبکه را بدهد:

```
# nmcli con add type ethernet con-name eth1 ifname enp0s3 ip4  
192.168.1.15/24 gw4 192.168.1.254
```

در این مثال، ما آدرس IP را به 192.168.1.15 /24 و روتر پیشفرض را به 192.168.1.254 تنظیم میکنیم.

## Traditional Command-Line Tools

میتوانید از یکی از چهار ابزار سنتی کامند لاین برای مدیریت شبکه استفاده کنید اگر توزیع لینوکس شما یکی از ابزارهای مدیریت شبکه Network Manager را پشتیبانی نکند:

تنظیمات اترنت برای یک رابط شبکه را نمایش می‌دهد. **ethtool**

. آدرس IP و مقادیر ماسک شبکه را برای یک رابط شبکه نمایش می‌دهد یا تنظیم می‌کند.

. آدرس IP، ماسک شبکه و مقادیر روتر را برای یک رابط شبکه نمایش می‌دهد یا تنظیم می‌کند.

. کلید رمزگذاری برای یک رابط بی‌سیم را نمایش می‌دهد یا تنظیم می‌کند.

. آدرس روتر پیش‌فرض را نمایش می‌دهد یا تنظیم می‌کند.

دستور ethtool به شما امکان می‌دهد تا ویژگی‌های کارت شبکه اترنت را بررسی کرده و تغییرات لازم را در تنظیمات برای ارتباط با دستگاه‌های شبکه مانند سوئیچ اعمال کنید. به طور پیش‌فرض، دستور ethtool تنظیمات پیکربندی فعلی رابط شبکه را نمایش می‌دهد، همانطور که در لیستینگ 15.2 نشان داده شده است.

Listing 15.2: Output from the ethtool command

```
$ ethtool enp0s3

Settings for enp0s3:

Supported ports: [ TP ]
Supported link modes:      10baseT/Half 10baseT/Full
                                         100baseT/Half
100baseT/Full                                         1000baseT/Full

Supported pause frame use: No
Supports auto-negotiation: Yes
Supported FEC modes: Not reported

Advertised link modes:      10baseT/Half 10baseT/Full
                                         100baseT/Half
100baseT/Full                                         1000baseT/Full

Advertised pause frame use: No
Advertised auto-negotiation: Yes
Advertised FEC modes: Not reported
```

```

Speed: 1000Mb/s
Duplex: Full
Port: Twisted Pair
PHYAD: 0
Transceiver: internal
Auto-negotiation: on
MDI-X: off (auto)

Cannot get wake-on-lan settings: Operation not permitted

Current message level: 0x00000007 (7)

drv probe link

Link detected: yes

```

می‌توانید ویژگی‌هایی مانند سرعت، دوپلکس و اینکه آیا رابط شبکه سعی می‌کند ویژگی‌ها را به طور خودکار با سوئیچ مذاکره کند یا خیر، با دستور `ethtool` تغییر دهید.

دستور `ifconfig` یک دستور `legacy` برای پیکربندی تنظیمات دستگاه‌های شبکه است. این دستور به شما امکان می‌دهد تا آدرس شبکه و ماسک را برای یک رابط شبکه تنظیم یا نمایش دهید:

```
$ sudo ifconfig enp0s3 down 10.0.2.10 netmask 255.255.255.0
```

دستور `ip` در انجام کارهای مختلف شبکه‌ای که می‌تواند انجام دهد، قدرتمندتر است و به عنوان روشی محبوب‌تر برای تعریف تنظیمات شبکه از کامند لاین شناخته می‌شود. ابزار `ip` از چندین گزینه دستور برای نمایش تنظیمات شبکه فعلی یا تعریف تنظیمات جدید استفاده می‌کند. جدول 15.1 این دستورها را نمایش می‌دهد.

Table 15.1 The ip utility command options

Description	Parameter
نمایش یا تنظیم آدرس IPv6 یا IPv4 بر روی دستگاه.	address
تعریف برچسب‌های پیکربندی	addrlabel

تунل اernetes از طریق IP	l2tp
یک دستگاه شبکه را تعریف می کند	link
یک آدرس چندگانه (Multicast address) برای سیستم برای گوش دادن به آن تعریف می کند.	maddress
پیامهای Netlink را نظارت می کند.	monitor
تعریف یک ورودی در حافظه Cache مسیریابی .Multicast	mroute
تعریف یک قانون در پایگاه داده Policy مسیریابی .Multicast	mrule
کشف cache یا ورودی های ARP با Manage Address Resolution Protocol همسایه (NDISC)	neighbor
مدیریت namespace های شبکه	netns
مدیریت عملیات cache همسایه.	ntable
مدیریت جدول مسیریابی.	route
مدیریت ورودی ها در پایگاه داده policy مسیریابی.	rule
مدیریت معیارهای TCP بر روی رابط.	tcpmetrics
مدیریت شناسه های رابط tokenized شده.	token
تунل از طریق IP.	tunnel
مدیریت دستگاه های تونل شبکه (TUN) یا (TAP) Network Bridge	tuntap
مدیریت سیاست های IPSec برای اتصالات امن.	Xfrm

هر گزینه دستور از پaramترها استفاده می کند تا مشخص کند چه کاری انجام دهد، مانند نمایش تنظیمات شبکه یا اصلاح تنظیمات شبکه موجود. لیست 15.3 نحوه نمایش تنظیمات شبکه فعلی را با استفاده از پaramتر show نشان می دهد.

```
$ ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

```

inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP group default qlen 1000
link/ether 08:00:27:73:1c:6d brd ff:ff:ff:ff:ff:ff
inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic
enp0s3
    valid_lft 84411sec preferred_lft 84411sec
inet6 fe80::5432:eddb:51ea:fb44/64 scope link noprefixroute
    valid_lft forever preferred_lft forever
$
```

لیست 15.3 دو رابط شبکه را در سیستم لینوکس نشان می‌دهد:

- lo – رابط loopback
- 3enp0s – یک رابط شبکه سیمی

رابط loopback محلی یک رابط شبکه مجازی ویژه است. هر برنامه محلی می‌تواند از آن استفاده کند تا با سایر برنامه‌ها مانند اینکه از طریق یک شبکه باشند، ارتباط برقرار کند. این می‌تواند فرآیند انتقال داده بین برنامه‌ها را ساده‌تر کند.

رابط شبکه 3enp0s، اتصال شبکه سیمی برای سیستم لینوکس است. دستور ip address add آدرس IP اختصاص داده شده به رابط (یک آدرس IPv6 و یک آدرس IPv4) و یک لوکل اختصاص داده شده است، مقدار netmask و برخی آمارهای اولیه درباره بسته‌ها بر روی رابط را نشان می‌دهد.

اگر خروجی نمایان‌گر یک آدرس شبکه اختصاص داده شده به رابط نباشد، می‌توانید از دستور ip برای مشخص کردن آدرس میزبان و مقادیر netmask برای رابط استفاده کنید:

```
# ip address add 10.0.2.15/24 dev enp0s3
```

سپس می‌توانید از دستور ip برای تنظیم روت‌ر پیش‌فرض برای رابط شبکه استفاده کنید:

```
# ip route add default via 192.168.1.254 dev enp0s3
```

و در آخر، اینترفیس شبکه را با استفاده از آپشن link فعال کنید:

```
# ip link set enp0s3 up
```

اگرچه دستور ip روشی جامع برای تغییر تنظیمات شبکه است، روش جایگزینی برای تعیین تنظیمات مسیریابی شبکه با استفاده از دستور route است:

```
# route add default gw 192.168.1.254
```

همچنین می‌توانید از دستور route به تنها‌ی استفاده کنید تا روت‌ر پیش‌فرض کنونی تنظیم شده برای سیستم را مشاهده کنید:

```
$ route
Kernel      IP      routing table
Destination      Gateway      Genmask      Flags
Metric    Ref    Use
Iface

default          192.168.1.254      0.0.0.0      UG
0            0        0

enp0s3
192.168.1.0      *      255.255.255.0      U
1            0        0

enp0s3
$
```

مسیریاب پیش‌فرض برای سیستم لینوکس 3enp0s 192.168.1.254 تعریف شده است و از رابط شبکه قابل دسترسی است. خروجی همچنین نشان می‌دهد که برای دسترسی به شبکه 192.168.1.0 به گیت‌وی نیاز نیست، زیرا این شبکه محلی است که سیستم لینوکس به آن متصل است.

اگر شبکه شما از طریق چندین شبکه و با استفاده از چندین روتر به شبکه‌ها متصل است، می‌توانید با استفاده از گزینه‌های کامند لاینی `route add` یا `route del`، جدول مسیریابی را به صورت دستی در سیستم ایجاد کنید. فرمات این دستور به صورت زیر است:

```
route [add] [del] target gw gateway
```

اگر شبکه‌ی شما از DHCP استفاده می‌کند، مطمئن شوید که یک برنامه کلاینت DHCP مناسب در سیستم لینوکس شما در حال اجرا است. برنامه کلاینت DHCP با سرور DHCP شبکه در پس زمینه ارتباط برقرار کرده و تنظیمات ضروری آدرس IP را بر اساس دستورات سرور DHCP اختصاص می‌دهد. سه برنامه کلاینت DHCP معمول برای سیستمهای لینوکس عبارتند از:

- `dhcpd`
- `dhclient`
- `pump`

برنامه `dhcpd` در حال تبدیل به محبوبترین بین این سه برنامه است، اما هنوز هم ممکن است از دو برنامه دیگر در برخی توزیع‌های لینوکس استفاده شود.

هنگام استفاده از ابزار مدیریت بسته‌های نرم‌افزار سیستم لینوکس خود برای نصب برنامه کلاینت DHCP، این برنامه به طور خودکار به زمان بوت سیستم راه‌اندازی می‌شود و پیکربندی آدرس IP مورد نیاز برای تعامل در شبکه را انجام می‌دهد.

قبل از استفاده از دستور `ip` برای اختصاص آدرس به یک رابط بی‌سیم، باید از طریق دستور `iwconfig` مقادیر SSID بی‌سیم و کلید رمزگذاری را تنظیم کنید:

```
# iwconfig wlan0 essid "MyNetwork" key s:mypassword
```

پارامتر `ssid` نام دسترسی به نقطه را مشخص می‌کند و پارامتر `key` کلید رمزگذاری مورد نیاز برای اتصال به آن را مشخص می‌کند. توجه کنید که کلید رمزگذاری با `s:` آغاز می‌شود. این به شما امکان می‌دهد که کلید رمزگذاری را با استفاده از کاراکترهای متن ASCII مشخص کنید – در غیر این صورت، باید کلید را با استفاده از مقادیر شش‌دهی مشخص کنید.

اگر نام یک اتصال بی‌سیم محلی را نمی‌دانید، می‌توانید از دستور `iwlist` برای نمایش همه سیگنال‌های بی‌سیم که کارت بی‌سیم شما شناسایی می‌کند استفاده کنید. فقط نام دستگاه بی‌سیم را مشخص کنید و از گزینه‌ی `scan` استفاده کنید:

```
$ iwlist wlan0 scan
```



### Obtaining Wi-Fi Drivers

متأسفانه، پشتیبانی از درایورهای سختافزار وایفای در لینوکس نسبتاً ضعیف است. اگر هنگام تنظیم وایفای، سختافزار وایفای خود را نمیبینید، ممکن است نیاز به پیدا کردن درایورهای مناسب داشته باشید. میتوانید با استفاده از ابزاری به نام lspci که در فصل 6، "Managing Hardware" توضیح داده شده است، این کار را شروع کنید. این دستور را بدون انتخابها اجرا کنید تا لیستی از سختافزارهای موجود را ببینید و در آن لیست برای یک آدپتور شبکه بیسیم جستجو کنید. به عنوان مثال، خروجی lspci لپتاپ من شامل خط زیر است:

```
03:00.0 Network controller: Realtek Semiconductor Co., Ltd.
```

```
RTL8191SEvB Wireless LAN Controller (rev 10)
```

این خط وایفای را به عنوان یک آدپتور Realtek RTL8191SEvB شناسایی میکند. جستجو در وبسایت Realtek به درایور منجر میشود؛ با این حال، این درایور باید به صورت محلی کامپایل شود که موضوعی است که خارج از دامنه این کتاب است.

همچنین ممکن است اتفاقاً در پیدا کردن یک درایور موفق نباشد. یک جایگزین برای استفاده از درایورهای لینوکسی محلی، استفاده از یک درایور ویندوز است. این گزینه غیرمعمول با استفاده از یک بسته به نام ndiswrapper (وبسایت <http://ndiswrapper.sourceforge.net>) امکان‌پذیر است که به شما اجازه می‌دهد تا درایورهای وایفای ویندوز را در لینوکس نصب کنید. همه توزیع‌ها ndiswrapper را در مجموعه‌های بسته استاندارد خود ارائه نمی‌دهند، اما معمولاً میتوانید یک بسته باینتری را در یک ریپازیتوری پیدا کنید.

اگر همه گزینه‌های دیگر شکست بخورند، ممکن است نیاز به خرید سختافزار شبکه جدید داشته باشید. بسیاری از آدپتورهای USB وایفای در دسترس هستند، اما باید آن‌ها را برای یافتن یکی که پشتیبانی خوبی از لینوکس دارد، تحقیق کنید. همچنین میتوانید آدپتورهای داخلی در برخی از لپتاپ‌ها را هم جایگزین کنید.

### Basic Network Troubleshooting

بعد از نصب کرنل لینوکس، میتوانید چند مرحله را انجام دهید تا اطمینان حاصل کنید که همه چیز به درستی کار می‌کند. این بخش شامل دستوراتی است که باید برای نظارت بر فعالیت شبکه بدانید، از جمله نظارت بر فرایندهایی که بر روی شبکه گوش می‌دهند و اتصالات فعلی از سیستم شما، می‌گذرد.

## Sending Test Packets

یک روش برای آزمایش اتصال شبکه ارسال بسته‌های آزمایشی به میزبان‌های شناخته شده است. لینوکس ICMP دستورات ping و 6ping را برای انجام این کار فراهم می‌کند. دستورات ping و 6ping بسته‌های پروتکل IP را با استفاده از پروتکل‌های IPv6 یا IPv4 به هاست‌های ریموت را ارسال می‌کنند. بسته‌های ICMP به صورت پشت صحنه برای ردیابی اتصالات و ارسال پیام‌های کنترلی بین سیستم‌ها کار می‌کنند. اگر هاست ریموت از ICMP پشتیبانی کند، هنگام دریافت یک بسته ping، یک پاسخ بر می‌گرداند.

فرمت پایه دستور ping به شرح زیر است که آدرس IP هاست ریموت را مشخص می‌کند:

```
$ ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2) 56(84) bytes of data.
64 bytes from 10.0.2.2: icmp_seq=1 ttl=63 time=14.6 ms
64 bytes from 10.0.2.2: icmp_seq=2 ttl=63 time=3.82 ms
64 bytes from 10.0.2.2: icmp_seq=3 ttl=63 time=2.05 ms
64 bytes from 10.0.2.2: icmp_seq=4 ttl=63 time=0.088 ms
64 bytes from 10.0.2.2: icmp_seq=5 ttl=63 time=3.54 ms
64 bytes from 10.0.2.2: icmp_seq=6 ttl=63 time=3.97 ms
64 bytes from 10.0.2.2: icmp_seq=7 ttl=63 time=0.040 ms
^C
--- 10.0.2.2 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6020ms
rtt min/avg/max/mdev = 0.040/4.030/14.696/4.620 ms
$
```

دستور ping به ارسال بسته‌ها ادامه می‌دهد تا زمانی که شما دکمه Ctrl+C را فشار دهید. همچنین می‌توانید از گزینه کامند لاین-c استفاده کنید تا تعداد مشخصی از بسته‌ها را برای ارسال مشخص کنید و سپس ارسال را متوقف کنید.

در دستور ping6، وضعیت‌ها کمی پیچیده‌تر می‌شوند. اگر از آدرس IPv6 لینک لوکال استفاده می‌کنید، نیاز دارد تا به دستور بگویید که بسته‌ها را از کدام رابط ارسال کند:

```

$ ping6 -c 4 fe80::c418:2ed0:aead:cbce%enp0s3
PING fe80::c418:2ed0:aead:cbce%enp0s3(fe80::c418:2ed0:aead:cbce) 56
data
bytes
64 bytes from fe80::c418:2ed0:aead:cbce: icmp_seq=1 ttl=128 time=1.47
ms
64 bytes from fe80::c418:2ed0:aead:cbce: icmp_seq=2 ttl=128
time=0.478 ms
64 bytes from fe80::c418:2ed0:aead:cbce: icmp_seq=3 ttl=128
time=0.777 ms
64 bytes from fe80::c418:2ed0:aead:cbce: icmp_seq=4 ttl=128
time=0.659 ms
--- fe80::c418:2ed0:aead:cbce%enp0s3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 0.478/0.847/1.475/0.378 ms
$
```

3% به سیستم می‌گوید که بسته‌های ping را از رابط شبکه enp0s3 برای آدرس لینک لوکال ارسال کند.

نکته: در این روزها، بسیاری از هاست‌ها از بسته‌های ICMP پشتیبانی نمی‌کنند زیرا می‌توانند برای ایجاد حمله DOS علیه هاست استفاده شوند. اگر سعی کردید یک هاست ریموت را ping کنید و پاسخی دریافت نکردید، متعجب نشوید.

## Finding Host Information

گاهی اوقات مشکل با اتصال شبکه نیست بلکه با سیستم نام میزبان DNS است. شما می‌توانید با استفاده از دستور host یک نام میزبان را تست کنید:

```
$ host www.linux.org
www.linux.org is an alias for linux.org.
linux.org has address 107.170.40.56
linux.org mail is handled by 20 mx.iqemail.net.
$
```

دستور host برای پرسش جوی سرور DNS برای تعیین آدرس‌های IP اختصاص داده شده به نام میزبان مشخص شده استفاده می‌شود. به طور پیش‌فرض، همه آدرس‌های IP مرتبط با نام میزبان برگردانده می‌شود. برخی از میزبان‌ها توسط چندین سرور در یک پیکربندی توازن باری پشتیبانی می‌شوند. دستور host تمام آدرس‌های IP مرتبط با این سرورها را نمایش می‌دهد.

```
$ host www.google.com
www.google.com has address 74.125.138.104
www.google.com has address 74.125.138.105
www.google.com has address 74.125.138.147
www.google.com has address 74.125.138.99
www.google.com has address 74.125.138.103
www.google.com has address 74.125.138.106
www.google.com has IPv6 address 2607:f8b0:4002:c0c::67
$
```

همچنین می‌توانید برای دستور host یک آدرس IP مشخص کنید و این دستور سعی می‌کند نام میزبان مرتبط با آن را پیدا کند:

```
$ host 107.170.40.56
56.40.170.107.in-addr.arpa domain name pointer iqdig11.iqnection.com.
$
```

به یاد داشته باشید که اغلب یک آدرس IP به یک نام هاست سرور تبدیل می‌شود که وبسایت را میزبانی می‌کند، نه نام جایگزین وبسایت، همانند آدرس آی پی www.linux.org

یک ابزار عالی دیگر دستور dig است. این دستور تمام رکوردهای داده DNS مرتبط با یک میزبان یا شبکه خاص را نمایش می‌دهد. به عنوان مثال، می‌توانید اطلاعات یک نام میزبان خاص را جستجو کنید:

```

$ dig www.linux.org

; <>> DiG 9.9.4‐RedHat&hyphen;9.9.4&hyphen;18.el7_1.5 <>>

www.linux.org

;; global options: +cmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45314

;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:

; EDNS: version: 0, flags:; udp: 4096

;; QUESTION SECTION:

;www.linux.org. IN A

;; ANSWER SECTION:

www.linux.org. 14400 IN CNAME linux.org.

linux.org. 3600 IN A 107.170.40.56

;; Query time: 75 msec

;; SERVER: 192.168.1.254#53(192.168.1.254)

;; WHEN: Sat Feb 06 17:44:29 EST 2016

;; MSG SIZE rcvd: 72

$
```

یا می توانید با استفاده از دستور dig، رکوردهای داده DNS مرتبط با یک سرویس خاص شبکه، مانند یک سرور ایمیل، را جستجو کنید:

```

$ dig linux.org MX

; <>> DiG 9.9.5<hyphen>3ubuntu0.5<hyphen>Ubuntu <>> linux.org MX

;; global options: +cmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16202

;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:

; EDNS: version: 0, flags:; udp: 4096

;; QUESTION SECTION:

;linux.org. IN MX

;; ANSWER SECTION:

linux.org. 3600 IN MX 20 mx.iqemail.net.

;; Query time: 75 msec

;; SERVER: 127.0.1.1#53(127.0.1.1)

;; WHEN: Tue Feb 09 12:35:43 EST 2016

;; MSG SIZE rcvd: 68

$
```

اگر نیاز دارید اطلاعات DNS مربوط به چندین سرور یا دامنه را جستجو کنید، دستور nslookup یک اینترفیس تعاملی فراهم می‌کند که در آن می‌توانید دستورات وارد کنید:

```

$ nslookup

> www.google.com

Server: 192.168.1.254

Address: 192.168.1.254#53

Non-authoritative answer:

Name: www.google.com
```

Address: 172.217.2.228

> www.wikipedia.org

Server: 192.168.1.254

Address: 192.168.1.254#53

Non-authoritative answer:

Name: www.wikipedia.org

Address: 208.80.153.224

> exit

\$

شما همچنین می‌توانید به صورت پویا آدرس یک سرور DNS دیگر را برای استفاده در جستجوهای نام مشخص کنید، که این یک راهکار مفید برای تعیین این است که آیا سرور DNS پیشفرض شما در صورت عدم موفقیت در حل نام مشکل دارد یا خیر.

## Advanced Network Troubleshooting

علاوه بر آزمایش‌های ساده شبکه که در بخش قبلی نشان داده شده‌اند، لینوکس دارای برنامه‌های پیشرفته‌تری است که می‌توانند اطلاعات پیچیده‌تری درباره محیط شبکه ارائه دهند. گاهی اوقات مفید است که بتوانید ببینید چه اتصالات شبکه‌ای در یک سیستم لینوکس فعال هستند. دو روش برای عیب‌یابی این موضوع وجود دارد: دستور netstat و دستور ss.

### The netstat Command

دستور netstat می‌تواند اطلاعات زیادی در مورد شبکه برای شما فراهم کند. به طور پیشفرض، تمامی اتصالات باز شبکه در سیستم را لیست می‌کند:

```
# netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address Foreign Address
State
```

Active UNIX domain sockets (w/o servers)

Proto	RefCnt	Flags	Type	State	I-Node
					Path
unix	2	[ ]	DGRAM		
	10825				
					@/org/freedesktop/systemd1/notify
unix	2	[ ]	DGRAM		
	10933				
					/run/systemd/shutdownd
unix	6	[ ]	DGRAM		
	6609				
					/run/systemd/journal/socket
unix	25	[ ]	DGRAM		
	6611	/dev/log			
unix	3	[ ]	STREAM	CONNECTED	25693
unix	3	[ ]	STREAM	CONNECTED	20770
					/var/run/dbus/system_bus_socket
unix	3	[ ]	STREAM	CONNECTED	19556
unix	3	[ ]	STREAM	CONNECTED	19511
unix	2	[ ]	DGRAM		
	24125				
unix	3	[ ]	STREAM	CONNECTED	19535
unix	3	[ ]	STREAM	CONNECTED	18067
					/var/run/dbus/system_bus_socket
unix	3	[ ]	STREAM	CONNECTED	32358
unix	3	[ ]	STREAM	CONNECTED	24818
					/var/run/dbus/system_bus_socket

...

دستور netstat خروجی زیادی تولید می‌کند، زیرا معمولاً چندین برنامه از سرویس‌های شبکه بر روی سیستم‌های لینوکس استفاده می‌کنند. شما می‌توانید خروجی را فقط به اتصالات TCP یا UDP محدود کنید با استفاده از گزینه کامند لاین †- برای اتصالات TCP یا u- برای اتصالات UDP:

```
$ netstat -t

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address           Foreign Address
State

tcp      1      0          10.0.2.15:58630
productsearch.ubuntu:https

CLOSE_WAIT

tcp6     1      0          ip6-localhost:57782 ip6-
localhost:ipp

CLOSE_WAIT

$
```

شما می‌توانید با استفاده از گزینه -l فهرستی از برنامه‌هایی که در کدام پورت‌های شبکه گوش می‌دهند، دریافت کنید:

```
$ netstat -l

Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
State

tcp      0      0      ubuntu02:domain     *:*
LISTEN
tcp      0      0      localhost:ipp       *:*
LISTEN
tcp6     0      0      ip6-localhost:ipp   [::]:*
LISTEN
udp      0      0      *:ipp              *:*
udp      0      0      *:mdns             *:*
udp      0      0      *:36355            *:*
udp      0      0      ubuntu02:domain     *:*
udp      0      0      *:bootpc           *:*
```

```

    udp      0      0  *:12461          *:*
    udp6     0      0  [::]:64294       [::]:*
    udp6     0      0  [::]:60259       [::]:*
    udp6     0      0  [::]:mdns        [::]:*
...

```

همانطور که می‌بینید، حتی یک ورک استیشن استاندارد لینوکس نیز همچنان کارهای زیادی در پس‌زمینه انجام می‌دهد و منتظر اتصالات است.

یکی دیگر از ویژگی‌های عالی دستور netstat این است که گزینه -s- آمارهای مربوط به انواع بسته‌های مختلفی که سیستم در شبکه استفاده کرده است را نمایش می‌دهد:

```
# netstat -s
```

Ip:

```

240762 total packets received
0 forwarded
0 incoming packets discarded
240747 incoming packets delivered
206940 requests sent out
32 dropped because of missing route

```

Icmp:

```

57 ICMP messages received
0 input ICMP message failed.

ICMP input histogram:

destination unreachable: 12
timeout in transit: 38
echo replies: 7

7 ICMP messages sent
0 ICMP messages failed

```

ICMP output histogram:

echo request: 7

IcmpMsg:

InType0: 7

InType3: 12

InType11: 38

OutType8: 7

Tcp:

286 active connections openings

0 passive connection openings

0 failed connection attempts

0 connection resets received

0 connections established

239933 segments received

206091 segments send out

0 segments retransmited

0 bad segments received.

0 resets sent

Udp:

757 packets received

0 packets to unknown port received.

0 packet receive errors

840 packets sent

0 receive buffer errors

```
0 send buffer errors
```

```
UdpLite:
```

```
TcpExt:
```

```
219 TCP sockets finished time wait in fast timer  
15 delayed acks sent  
26 delayed acks further delayed because of locked socket  
Quick ack mode was activated 1 times  
229343 packet headers predicted  
289 acknowledgments not containing data payload received  
301 predicted acknowledgments  
TCPRecvCoalesce: 72755
```

```
IpExt:
```

```
InNoRoutes: 2  
InMcastPkts: 13  
OutMcastPkts: 15  
InOctets: 410722578  
OutOctets: 8363083  
InMcastOctets: 2746  
OutMcastOctets: 2826
```

```
#
```

خروجی آمارهای netstat می‌تواند به شما ایده‌ای تقریبی از میزان مشغولیت سیستم لینوکس شما در شبکه بدهد، یا اگر مشکلی در یکی از پروتکل‌های نصب شده وجود دارد، نشان دهد.

## Examining Sockets

ابزار netstat اطلاعات شبکه‌ای زیادی فراهم می‌کند، اما اغلب ممکن است سخت باشد که تشخیص دهیم کدام برنامه در کدام پورت باز گوش می‌دهد. دستور ss می‌تواند در این زمینه کمک کند.

یک اتصال برنامه به یک پورت، یک سوکت نامیده می‌شود. دستور ss می‌تواند لینک کند که کدام فرآیندهای سیستم از کدام سوکتهای شبکه فعال استفاده می‌کنند:

```
$ ss -anpt
State Recv-Q Send-Q Local Address:Port Peer
Address:Port
LISTEN 0      100    127.0.0.1:25        *:*
LISTEN 0      128     *:111           *:*
LISTEN 0      5      192.168.122.1:53   *:*
LISTEN 0      128     *:22            *:*
LISTEN 0      128    127.0.0.1:631    *:*
LISTEN 0      100     ::1:25          :::*
LISTEN 0      128     ::111           :::*
LISTEN 0      128     ::22            :::*
LISTEN 0      128     ::1:631         :::*
ESTAB 0       0      ::1:22          ::1:40490
ESTAB 0       0      ::1:40490       ::1:22
users:(("ssh",pid=15176,fd=3))
$
```

گزینه `anpt` هم اتصالات TCP گوشده‌نه و هم اتصالات TCP برقرار شده را نمایش می‌دهد، و همچنین فرآیندی که با آن‌ها مرتبط است را نشان می‌دهد. این خروجی نشان می‌دهد که پورت ssh (پورت 22) یک اتصال برقرار دارد و توسط شناسه فرآیند 15176، برنامه ssh کنترل می‌شود.