LINUX

LEARN LINUX WITH EASE

FROM BEGINNER TO EXPERT
IN LESS THAN A WEEK

STEPHEN BLUMENTHAL

LINUX

From Beginner To Expert In Less Than A Week

By

Stephen Blumenthal

TABLE OF CONTENTS

INTRODUCTION

The Linux system often intimidates beginners, because of the need to type in a command or two every now and then.However, Linux commands (or shell commands, to put it more accurately) follow a rather logical and sensible pattern and all you need to do is get used to it! Most shell commands are roughly 3, or at most 4 letters long. The more fre q uently needed commands are shortened further usually.

The **man** command is a reference to 'manual', i.e. the documentation that comes preloaded in Linux systems to help you with common problems faced by a user. These man(ual) pages are certainly not meant to be standalone guides for beginners, but instead, they need to be used as reference material to corroborate what you've learnt/tried elsewhere.

The **info** command is another way to look up reference material for GNU information and troubleshooting suggestions. It isn't completely the same as the man pages, as there is a scope for using 'hyperlinks', to make it easier to browse through the material you need. There are shortcuts for scrolling through pages too.

The **date** command is self-explanatory. It will tell you about the current time and date on the system. On those lines, the cal command will display a neat calendar of the current month (or that of any particular one you choose).

The **ls** command is meant to help you list the contents of the directory you choose. This means that you will get to see all the files and directories inside your current directory.This command also has switches to modify its behavior and functionality. For example, adding -l after ls will give you a more detailed description of the contents of the directory you're in. you'll get to see details about file permissions, modification dates, groups, sizes and owners. The -a switch helps you see all the files in your current directory, including the hidden ones. (Hidden files in Linux have their filenames starting with a '.' i.e. a period)

In case you lose track of the current directory you are in, type pwd, which prints the current directory you are working in. Once you've got to know that, you can navigate around the Linux file system using the **cd** command, where 'cd' stands for 'change directory'. This can work on both relative and absolute terms, i.e., you can work by specifying the entire directory, starting from '/' (the home directory) to the directory/file you want to go to. Otherwise, you can also use the '.' or '..' system. The latter tells the compiler that you want to go to the parent directory of your current one, making it an easy alternative to typing the entire path every time.

When you start working with simple files, like text files, the cat command will come in handy. It 'concatenates' the content of one or more text files, letting you see all of the text they contain.

The ps command lists all the processes currently running on your system in your current terminal.

The shell commands in this page are a good way to start off in the Linux environment. They are simple enough for you to get an idea of how the entire system works, and will probably help you to try out more advanced steps later!

How To Get Started

Basic Linux For Ease of Use and Management of a Hosted Website - Getting Started!

Welcome to the beginning of a path to simpler website administration!

This chapter is the first in a series aimed at the average user with the average hosted website; someone without a vast amount of hyper-technical knowledge about the guts of the internet. We will see that utilizing the more "complicated" part of the hosting provider's service can be easy and can make your website management easier as well. I will provide you with a basic road map to simple Linux utilization that, I believe, will both benefit, please and, ultimately, empower you to better manage and amplify your site's potential!

As we all know, with the proliferation of user-hosted websites in our age, the market has driven most hosting services to provide users with more user-friendly interfaces; windows, or browser, based menuing systems with simple and easy to understand layouts.

This is a good thing, but it's also a bad thing. One could make the argument that the simplification of hosted website management through the use of these interfaces makes it possible for the average person to easily set up and manage their own website or e-business. This is true. In fact, it's one of the main reasons that the web hosting industry is experiencing a boom. Consumers demand ease-of-use from merchants and simple picture-menu based setup and maintenance systems fulfill that demand.

So how can this be a bad thing?

There are a multitude of reasons, but the one I'll be addressing here is, coincidentally, the exact same thing: Ease Of Use! What most people don't realize at first (but soon become aware of) is that the "Ease Of Use" supplied by these point-and-click-simple solutions, more often than not, make a lot of simple tasks extremely difficult, if not impossible!

The double-whammy is that they essentially "mystify" the underlying operating system (that place where the work actually gets done) and lock most people into a cycle of endless forum-searching or email-tag with customer support (and experienced Linux users) where they find, mostly, answers that assume they already possess whatever basic knowledge they need to complete the task at hand. I'm not saying that any of these folks are unfriendly or unwilling to help, just that the average user ends up no better off in the end and has to go back to... the windows-like menuing system.

The good news is: You don't have to settle!

Given the right circumstances, you can enhance your ability to maximize the potential of your website, and, ironically, increase your "Ease Of Use" by making use of that underlying operating system.

By learning just a few of the basics of Linux (the most popular operating system for most web-hosting solution providers), you can dramatically increase the ease with which you can maintain, and even optimize, your website.

Below, we'll go Point-by-Point, with "Tips" along the way, toward adding use of the Linux operating system to your bag of tricks. The de-mystification begins!

Point Number 1: Don't let any roadblocks you hit discourage you. What you're about to accomplish is within the realm of absolute possibility. You CAN use Linux to maintain your website, enhance it, optimize it and much more. None of this is beyond your ability. Societally perpetuated self-doubt, mass-belief that it's all just too complicated and the pervasive notion that Linux is comprehensible only to computer science majors and "techies" are myth's and falsehoods.

You can master Linux as easily as you can master your ABC's. If that seems like over-simplification, believe me when I tell you that it's not. Remember how much easier it was to remember LMNOP than it was the rest of the alphabet? In simplistic terms, you'll be learning how to help yourself by learning how to use Linux to your advantage in much the same way. We'll start with the parts that come easily, and the rest will fall into place over time, seemingly without effort!

Point Number 2: In order to make any of this work, your service provider will have to offer you the option of using a "shell account." This is the most common terminology for direct access to the underlying operating system. "Shell access" is also commonly understood.If you are already being hosted, be sure to use these terms when re q uesting the access you'll need. All service providers understand what a "shell account," or "shell access" is and will be able to let you know, immediately, whether or not they offer their users that option.

Tip: If you are in the consideration phase, and looking for a hosting provider, be sure to ask them if they offer you the option of a "shell account," or "shell access." It's your call in the end, but, if they don't, I would advise that you continue your search for a provider elsewhere.

Tip: Most providers offer "shell accounts," or "shell access," but they don't make a point of letting you know. In my experience, it's never anywhere near the top of the list of features the hosting provider offers, and, most times, you have to go to the support page, or elsewhere, to find out. You just need to ask. If it's something they offer, they'll give it to you (however grudgingly); usually with simple login instructions. Hosting providers generally don't like the thought of "regular" users mucking about with the underlying operating system, so they generally don't make it a point to let you know you have this option!

Point Number 3: Now you're going to need to get to your shell account. This is a piece of the puzzle that most service providers will assume you know how to do. We'll assume for the moment that you don't. For our purposes here we'll assume your provider is a company named XYZ.com and you already have access to the internet and have that connection active when you connect to your "shell account."

There are certain things that you're going to need in order to access your shell account; all of which are free. Some you'll have to get yourself and some you'll most likely have to re q uest from your service provider.

Most importantly, you'll need your connection information. You will get this from your service provider. It should include:

1. The host name or IP address of the server you'll need to connect to in order to access your account (e.g. webhost.XYZ.com or somecrazyname.XYZ.com). You'll almost never be given an IP address exclusively. Your service provider should, however, include this information along with the server name (e.g. webhost.XYZ.com - IP Address: 192.68.224.176). Having an IP address to connect to can be advantageous if, for some reason, you can't reach the server via the host name.

2. Your login information. This will simply be a user name and password.

3. The method by which you can access your shell account. Generally this will be via "SSH" (Secure Shell), but some hosts still use "Telnet" (Telecommunication Network

't let the definitions I've included in parentheses put you off. They're simply provided for completeness and shouldn't concern you at this point. Their strict "definitions" may never ever concern you - They don't concern me and I've been in the business for well over a decade!

Basically, the difference between the two connection methods is academic. Telnet sends information over the internet as-it-is. This is one of the reasons most providers use SSH. SSH sends information over the internet in "encrypted" form. That is, Telnet is an "unsecure" protocol, while SSH is considered "secure." It's much harder (if not nearly impossible) for someone to hack into your connection and "see" what you're typing if the information is encrypted. Encrypted information is protected. Unencrypted information (what you'll be sending if you use Telnet) can be read (by the proper hacker) as if he or she were looking over your shoulder watching you type!

Tip: Don't accept Telnet if it's offered as a connection option. Insist on SSH. If SSH is not available from your provider, there are other options you can pursue, but they're beyond the scope of this article.

Next, you'll need a method by which to connect, using the information given to you by your hosting provider. This is simply going to be some software "client" that you'll use to connect. Many SSH clients are available for free and can be downloaded at various freeware sites on the internet (Use any search engine and simply type in a search for "SSH client freeware download." You'll be surprised at the number of options available!)

Tip: Don't pay for an SSH client unless it makes you feel better. There are several reputable and highly effective clients available for free. Almost all work right out of the box (just start them up and look for a button that says "new connection," or something similar, and then you'll be presented with a screen into which you can type in the host name, user name and password information you received from your hosting provider. Just click connect and you're logging into your "shell account!"

Point Number 4: Now, strangely enough (with most providers), you'll be presented with a "menu screen" once you login to your "shell account." This will generally provide you with several options such as editing files, sending email, uploading or downloading files, etc.

A text-based menu is generally fairly easy to follow. Options are presented on a numbered menu (possibly with letter shortcuts in parentheses alongside), you select the number (or letter) of the option you want to use and then you do whatever that is until you exit and come back to the menu. A simple menu might look like this:

```
---- Welcome to XYZ.com Shell Access Menu -----

1. Edit Files (e)

2. Send Email (s)

3. Upload Files (u)
```

4. Download Files (d)

5. Linux Shell (l)

6. Quit (q)

Enter your option

Tip: If you use any option and it isn't made obvious how you can get back to the menuing system, you can generally get back there by "killing" whatever program you've launched by selecting your option. This canusually be done by typing one of the following "control-character" se q uences. ctl+c, ctl+x, ctl+v, ctl+d. The key combinations described here are simply the typing of two keys at once (denoted by the + symbol), so for ctl+c you would type the "control" key (usually "ctl" or "ctrl" on your keyboard) while simultaneously typing the "c" key. Just type them both at the same time. Nothing to it!

Please note that all of these options may disconnect you completely from your server and should be used only after you've saved any work you're doing.

Not to worry; if you do get disconnected, all you need to do is connect again. Of course, any and/or all of these options may do nothing at all. If you just "need" to disconnect and can find no remedy in your "shell account," you can always take the guaranteed step of closing your SSH client.

Now you have arrived!

This part of the lesson is coming to an end, but your journey has just begun. At this point, fool around with the various menu options and try out the various features of your "shell account." Use them with caution, as you would when interfacing with your window-based menuing system. Try to keep your actions non-destructive (e.g. If you're going to edit a file in a foreign editor, make sure to back it up, or copy if off, first, etc).

If your hosting provider's "shell account" is literally that, you'll end up at a "shell prompt" after connecting. Take some time to investigate. For now, stick to using "info," "help" or "man" (for manual) commands to learn about your environment. You'll know you're at a shell prompt immediately. It may look something like this (But, there's no mistaking it for a menu!)

```
/home/user/public_html >_
```

Practice with the skills you've gained so far. You'll find, with time that they will become second nature. Of course, we're only part of the way there now, but, as this article is a "Beginning," you have accomplished your goal.

Remember, with practice and patience, you can learn a thing or two about Linux as you explore your new environment. Be cautious, but have fun. It's one of the best ways to learn!

Getting A Better Understanding Of Linux

Understanding the Linux File System

The way Linux organizes its files on the hard drive is vastly different from how Windows handles this task. New Linux users coming from Windows sometime have a hard time maneuvering though directories or come with notions that Linux should manage its files in the same vain as Windows.

This article was written to help new users get a grasp on moving through directories on their new installation. One key point to make is Windows deals with "drives" as in your C: drive or D: drive, Linux deals with something called 'mount points'. These are locations where other hard drives, CD/DVD burners, etc... connect to the root partition. Don't worry it will all make sense latter on.

It All Begins With Root: /

The root directory known simply as '/' is the starting point. Without getting to technical, the root directory acts like the 'C: Drive' in Windows. A Linux system can not fully boot without a root partition, in the same way as deleting your C:WINDOWS folder will make your Windows computer inoperable.

It's In: /bin

The /bin folder holds important system programs. The 'bin' is short for 'binary'. Some of the popular programs: date, less, more, cat, dmesg. These programs are essential in order to start and have a complete operating system. While you may never use one of these programs personally, the system relies on some of them.

Where Everything Starts: /boot

As the name implies, /boot is where the crucial files reside, mainly the kernel. Without the kernel, you don't have a system. Another crucial program located in /boot is the bootloader. Just like Linux needs the kernel to function. The bootloader is there to actually locate the kernel and begin running it.

Every Device Is A File: /dev

In Linux, every device is a file. What this means is, when you connect a hard drive to your system it gets a 'device file' that allows the system to interact with it. When the kernel locates a new hard drive it is assigned a file like "/dev/sda". The /dev part is the directory and the 'sda' part is the file that connects to the hardware. So if you wanted to format your whole drive you could type in the command 'dd if=/dev/null of=/dev/sda'. This would copy /dev/null into your hard drive. /dev/null is a "bit bucket". Meaning that everything that gets sent to it gets deleted.

Configuration-ness: /etc

Linux, being a customizable system keeps all the programs config' files in this directory. Most programs come with a sensible and secure default behavior. But what happens if you want to change it? The /etc holds a slew of text files for you to open and customize how your programs operate. An important note to make is /etc manages global defaults. What this means is if you change a file this directory, it will affect the whole system.

The Shared Libraries: /lib

The /lib directory is a way to keep all software libraries in one central location. Most (if not all) files here have a file extension of '.so' to let you know they are 'shared object' files. These files are code that can be used by multiple programs. This helps prevent a problem known as 'software bloat'. Windows also has these files; they are called 'Dynamically Linked Libraries' or DLL for short. As a regular Linux end user, you will most likely never have to change anything in this folder. Depending on how you install software on your computer, you might come across a 'missing shared object' problem if your software "depends" on another program to function. The Windows e q uivalent is 'DLL hell'.

When You Don't Shut Down Correctly: /lost+found

This directory is used when the user does not shut down the system correctly (turning it off when the system is still up and running). Upon the next boot, the system will try and correct itself by scanning the hard drive for corrupt files and try to correct any problems that arise. If anything is found, it will be placed in the /lost+found directory for the systems administrator (you!) to see and look over.

Where The 'Mount Points' Live: /mnt and /media

The /mnt and /media directories are for 'attaching' other devices to the root directory. In Windows, when you insert a USB thumbstick, you will see the system gives it a drive letter (E:). Depending on which Linux distribution you use, the device will either 'auto mount' or the user has to mount the device manually. Most newer, newbie friendly distros will auto mount the device and place it in one of these directories. You will be able to browse the files within your thumbstick at /mnt/usb or /media/usb. Each distribution is different, so my example could not exactly match your results.

/media is the newcomer to the Linux scene. Most older distributions exclusively used /mnt to manage these devices, but /media is gaining ground as the default location to mount devices. Linux allows you to mount any device anywhere (as long as you have the permissions). So it is completely feasible to mount one device under '/bin/mount' or '/var/log'. This is usually not a good idea and the /mnt and /media directories where put in place to make this easier.

The 'Optional' Directory: /opt

This is where users can install software if no other suitable location can be used. Most software from major Linux distributions have 'software repositories' which allow users to easily add and remove tons of programs. But what happens when you need a program that isn't in the repository? In order to separate repository software packages from 'external' packages, sometimes the best way to install them is putting them in /opt. This practice is rarely used though and each distribution is different. Some will place the popular KDE into /opt, while other distributions won't.

My personal rule of thumb is to use /opt when the software you are installing defaults to this directory (The Google Earth program does this) or I am installing a program that I didn't get in the software repository.

The Kernel's Directory: /proc and /sys

Both of these directories hold a wealth of information about the status of your system. Files like '/proc/cpuinfo' contain information about your CPU (speed, vendor, cache size). The /proc directory is slowly being faded out in favor of /sys.

You Were Here And Now Your Gone: /tmp

The /tmp directory is short for 'temporary'. So with that in mind, I am sure you can deduce why this directory is used. You got it, to manage temporary files. Programs can generate a lot of 'junk output' or need to write to a file to handle a task; but the file can be deleted once the task is completed. This directory provides a central location to do this and not fill your other directories with these files.

Where The Programs Live: /usr

The /usr directory is a monster. Articles could be written just to explain it all. But to keep things short and sweet, the /usr is where all of your 'secondary' programs are stored. Granted you love your music player, but it's not crucial to your operating system actually functioning. So instead of putting all the executables in /bin, we break it up a bit.We place crucial system programs in /bin and non-critical programs into /usr/bin. The /usr directory could be seen as the Windows e q uivalent as C:Program Files.

The Not So Temporary Files: /var

/var (for varying or variable) acts like /tmp in the sense that the files located are 'temporary' but less 'temporary' then those in /tmp. What this really means is the /tmp directory will most likely be deleted every time the system reboots, while the files in /var will not. /var is a place to keep 'persistent' files. An example would be log files. Most system administrators wouldn't want to delete their log files on every reboot, but the files could be removed or 'shrunk' to a more manageable level at the administrators whim.

Another example would be '/var/mail' directory. It contains the mail being sent to users on the system. Some users will have hundreds of messages, while other users will have a few or none. The directory is growing and shrinking depending on the usage by the users.So in order to keep the disk usage under manageable levels, we place this activity under /var. On large systems, the system administrator will use a separate hard drive and 'mount' the hard drive at /var. This allows the fre q uent disk access to remain on one hard drive and keep the overall system speedy.

HOW TO INSTALL LINUX

Linux Training - Linux Installation Help - How To Install the Linux OS from a New Linux Download

You can install Linux from CD or DVD onto a new computer system. This will allow you to learn how to use Linux to get real, practical Linux training and experience. In this article, we're talking about a new system that doesn't already have an operating system (like Windows) on it.

You can download the Linux OS (operating system) as a Linux ISO file and burn it to CD or DVD yourself, or you can order Linux on CD or DVD and have it mailed to you.

Linux Tips: Linux ISO files are large. Only download Linux if you have high-speed Internet access. You can buy and have the Linux OS delivered by to you by mail very cheaply. Just do an Internet search for "list of linux cds" or "list of linux dvds".

Linux Tips: Linux DVDs hold a lot more programs than CDs. Order a Linux DVD version and you will have more Linux software programs to choose from than on a CD version - and you will also need fewer Linux DVDs than CDs!

7 Steps to Install Linux on a New Computer System

1. Document Your Linux Installation Settings

During the Linux installation, you need to specify some system settings. These include: the Linux software programs and desktop(s) you want installed, networking settings, and disk partition sizes.

2. Boot with the Linux CD or DVD and Start the Linux Installation Routine

Do the steps to set up your system to boot from its CD or DVD drive.

Shut down your system and boot it with the first Linux CD / DVD in the drive and start the Linux installation routine.

3. Specify Your Linux Installation Settings

Follow the on-screen prompts and put in the installation settings you documented in Step 1.

4. Create a Regular ("Non-root") User

You log in to work on a Linux system as a Linux "user", with a user name and password. You can log in as the "root" user, or as a "non-root" user.

You log in and work on the Linux OS as the root user when you need to do Linux system administration tasks. For example, you work as the root user to run a Linux command to create a new Linux user, or to do the steps to install a Linux server.

When you install Linux, the root user is always created automatically for doing Linux system administration tasks. However, for security reasons, you should never log in to a Linux desktop as the root user.

At some point during the installation routine, you will be asked if you want to create one or more "regular" (non-root)

Linux users. Always create at least one regular Linux user and log in as this user to do day-to-day tasks.

5. Let the Linux OS Install on Your Hard Disk

After you specify the Linux installation settings and create one or more new Linux users, the installation routine will copy the Linux OS and Linux software programs onto the hard disk in the system, and then reboot.

6. Log In to Test the System

Once the system reboots, log in as a "regular" (non-root) Linux user to test that you can do work as this user. At this point, the Linux desktop appears and you can run Linux programs to test the system.

Linux Tips: Remember not to log in to a Linux desktop as the root user.

Test the root user by logging in as a regular Linux user and opening a terminal emulation window. Then run the su command to log in and work as the root user.

7. Have fun!

The Linux OS is an amazing and extremely reliable system. And there are thousands of Linux software programs for all kinds of uses.

By installing and running Linux you can get lots of great practical Linux training experience while working with Linux. You can work at a Linux desktop and run commonly used Linux software programs. You can also work at the Linux command line and learn how to use Linux commands - the way the real pros do Linux system administration.

Linux Training - Linux Installation Help - Linux Live CDs & Linux Installation CDs - Pros and Cons

You get real, practical Linux training when you get Linux running on a computer system, and you work with it to get experience.

Once you get the Linux OS (operating system) running, you can learn how to use Linux desktops, run Linux software programs, and most importantly, go to the Linux command line to run Linux commands - the real power behind Linux system administration.

Two popular methods of getting to work with the Linux OS are:

1. Install Linux from CD or DVD onto the hard disk in a system

2. Boot and run Linux from Linux live CDs or DVDs (without installing Linux)

Here are some of the pros and cons of installing Linux on a system and running Linux from a live CD or DVD.

Linux Live CDs - Pros

1. The Linux OS boots and runs in just a few minutes entirely from a single Linux live CD (or DVD).

2. A lengthy Linux installation is not required. You don't need to provide a lot of settings to the Linux installation routine and then wait while the Linux OS and Linux software programs are copied onto the hard disk.

Also, if you install Linux on Windows, you need to back up your system before you do and you may damage your Windows setup during the Linux installation - or if you remove Linux later!

3. You only need to get 1 Linux live CD or DVD, as opposed to possibly needing several when you install Linux. Therefore, if you download Linux to burn a Linux ISO file to CD / DVD, you only need to wait while 1 Linux CD / DVD is downloaded.

4. Linux live CDs / DVDs are very inexpensive. If you don't want to download Linux as a live CD / DVD, you can order one and have it mailed to any location on the planet for a very small amount of money. Just run your browser and do a search for "list of linux cds" or "list of linux dvds".

Linux Tips: If you are going to buy a Linux live version and you have a DVD drive, get a Linux live DVD rather than a CD. A DVD holds about six times as much as a CD. Therefore, you'll get lots (and lots) more Linux software programs on a DVD!

5. A Linux live CD can easily be removed and taken anywhere. Portable Linux!

6. You can save your work (data files) - if you get the right Linux live CD or DVD.

Linux Tips: Before getting a Linux live CD / DVD, do some research to make sure you can save your work. Some Linux live distros let you save your data and some don't. For example, the Ubuntu Linux live CD allows you to save your work to a USB flash drive.

Linux Live CDs - Cons

1. Although booting from a Linux live CD / DVD is faster than doing a complete Linux installation, you still need to wait while your system boots from the live CD / DVD, every time your system starts. If you usually keep your system turned on, this isn't an issue.

2. You may not be able to save your work. Some Linux live CD / DVD distros allow you to save your work and some don't.

3. You need to boot from the Linux live CD / DVD every time your system starts. This may not be very convenient if you need to put other CDs or DVDs in the drive while running Linux "live". Any time you need to put something in your drive, you have to remove the live CD / DVD. This also increases the likelihood that your live CD or DVD will become damaged.

Linux Tips: Keep a copy of your current Linux live CD handy in case your working copy becomes damaged.

Either way, installing Linux or using Linux live CDs (or Linux live DVDs), is an excellent way to get Linux training so you can learn how to use Linux.

First Steps With Linux

Trying Linux for the First Time: A Beginner's Guide

You're probably familiar with Windows and/or Mac OS. But they aren't the only operating systems available. A popular alternative is Linux. In this chapter, Lesley Lutomski introduces Linux and what you need to know to give it a try.

I'm constantly surprised by people who tell me they'd like to try Linux, but think it's "too hard".

There seems to be a common misapprehension that Linux is "for geeks". Certainly, this was once the case: dedicated users compiled their own kernels, and it wasn't for the faint-hearted.

But Linux has come a long way since those days. So, if you've never tried it, or tried it many years ago and gave up, I'd encourage you to think again.

Choose Your Flavor

Linux comes in many "flavors", or "distributions"—normally referred to as distros. Some of these are aimed firmly at a mainstream audience, and I'd suggest using one of these to get your feet wet. The best known of these is possibly Ubuntu, which is the one I use and the one I'll concentrate on here. Linux Mint is also popular, but there are many more.

Difficult or Not?

So is it difficult to use? Not in my experience. The first thing I noticed when we switched to Ubuntu was the sudden reduction in the number of distress calls I received from my husband. He seemed to experience fewer problems using the system than he had on Windows XP, and also seemed to feel more confident about trying things for himself, rather than panicking that he might “break something”.

I also set up a Xubuntu system for an elderly friend who had never used a computer of any kind, and she rapidly got to grips with it.

What Are the Benefits?

For many people, cost will be a consideration. Most popular Linux distros—and their associated software—are free to download and use. For others, the open-source nature of the OS appeals.

Linux is also far less susceptible to viruses than Windows. The main reason for this is simply that most viruses are designed to target Windows machines and will have no effect on a Linux system. It's not true that Linux systems are immune to viruses, but they are very rare. A humorous explanation can be found here.

This added security is one reason we chose it for our elderly friend. Although Linux viruses are rare, ClamAV is free and helps ensure you don't inadvertently download and pass on viruses to friends with Windows.

Will Linux Be Compatible with My Hardware?

Linux will run well on most PCs, although if you have the very latest cutting-edge technology, you may find it's not immediately supported.

On the other hand, installing Linux can be a great way to breathe new life into old hardware. Some distros are designed to be lightweight—such as the Ubuntu variant Xubuntu—and will perform well on systems with limited resources.

It's also possible to run Linux on a Mac, although I have no experience of this. The Ubuntu Forums—a great source of help and support—have a dedicated section for Apple hardware users.

How to Choose a Distro?

The easiest way is simply to try one and see if you like it. This isn't nearly as radical as it sounds.

Many distros are free to download, after which you can burn them to DVD. They can then be run as a “live” CD/DVD. In other words, you boot your system from the DVD—or a USB drive—and run the OS from there. It doesn't have to be installed, and nothing is written to your hard drive—although you should be able to access files on your hard drive while in Linux.

This is a great way to get a feel for the distro at no risk, and it also lets you check there are no problems with your hardware. The Ubuntu site provides very clear tutorials for getting all this done.

How Do I Install It?

Again, this is a simple process. The “live” CD/DVD includes options to install the system, should you choose to do so, and will walk you through the process.

You can choose to install alongside another OS—either on the same drive or a separate drive—and run a dual-boot system. This allows you to use either system. You can also run it in a VM.

If you decide to install Ubuntu or its variants, either as a dual-boot or standalone system, the process is as simple as answering a few questions and then sitting back while the system does the work. If you're installing alongside an existing OS on the same drive, Ubuntu will take care of partitioning it. If you have a broadband connection, you can choose to download and install updates as part of the process.

Be warned that Linux uses a different file system to Windows. While Linux will be able to read and write to a Windows partition, Windows won’t read the Linux partition. If you’re dual-booting and need to access files from both systems, ensure you save them to a partition formatted as FAT or NTFS.

My own experience of this as a first-time Linux user was not q uite straightforward. I added a second hard drive for Ubuntu and the installation went smoothly. Afterwards, I booted into Ubuntu and everything was great—until I tried to boot WinXP and found I couldn't.

After a brief panic, I headed off to the Ubuntu Forums, where somebody patiently walked me through a couple of possible solutions until we found one which worked for my system. That was seven years ago, and the only time I had an issue like that.

What About Other Software?

This is probably the ultimate determinant of whether or not Linux is for you. On the one hand, you have access to a great deal of free, easy-to-install software; on the other, many popular commercial applications are not available for Linux.

You can, of course, run a VM, or an alternative is to use Wine, which is a kind of translation layer for Windows software. The Wine website maintains an Application Database that gives guidance on how well each application runs under Wine. Often this can vary greatly, depending on the version of the software, as in these results for Photoshop:

I've had good success with old games and smaller apps, but generally speaking, I use native Linux alternatives.

LibreOffice (a fork of OpenOffice) comes installed as standard, as does Firefox. Chromium is the native Linux version of the Chrome browser and supports many of the same extensions. GIMP is a near replacement for Photoshop, although opinions vary on how well it compares. In addition to the bundled applications, other software is easy to install from the repositories.

Ubuntu now comes with the Ubuntu Software Centre, which is a graphical interface that lets you find software by category

I tend to prefer the older Synaptic Package Manager. You can search for an application by name or keyword.

What About Updates?

Updates are notified automatically and installed with a single click. It's rare to have to restart the system, except for updates to the kernel itself.

New releases of Ubuntu are made every six months, but you can opt to use a “Long Term Support” (LTS) version. These are released every two years. Installed applications will receive updates, but will generally not upgrade to a newer version until the next Ubuntu release. So if you like to run the latest version of things, the LTS version won't be for you.

So What Are You Waiting For?

This was never intended as a “how to” guide, but just as an overview to whet your appetite and encourage you to try Linux for yourself. Hopefully I've succeeded, and you're now off to download your very first distro!

If you'd rather dip your toe in an even gentler way, without even burning and booting from a DVD, the Ubuntu site offers a virtual tour of the operating system that you might enjoy. It gives you a taste of the look and feel of Ubuntu, including a glimpse of the main apps that come with it.

Lastly, if you’ve already taken the plunge and installed Linux, and you’re a web developer, here’s some advice on setting up a development environment in Linux.

Post-Installation Activities

This section contains optional procedures for configuring Linux hosts to work better with Docker.

Manage Docker as a non-root user

The docker daemon binds to a Unix socket instead of a TCP port. By default that Unix socket is owned by the user root and other users can only access it using sudo. The docker daemon always runs as the root user.

If you don't want to use sudo when you use the docker command, create a Unix group called docker and add users to it. When the docker daemon starts, it makes the ownership of the Unix socket read/writable by the docker group.

Warning: The docker group grants privileges e q uivalent to the root user. For details on how this impacts security in your system, see Docker Daemon Attack Surface.

To create the docker group and add your user:

Create the docker group.

```
$ sudo groupadd docker
```

Add your user to the docker group.

```
$ sudo usermod -aG docker $USER
```

Log out and log back in so that your group membership is re-evaluated.

If testing on a virtual machine, it may be necessary to restart the virtual machine for changes to take affect.

On a desktop Linux environment such as X Windows, log out of your session completely and then log back in.

Verify that you can run docker commands without sudo.

```
$ docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints an informational message and exits.

Configure Docker to start on boot

Most current Linux distributions (RHEL, CentOS, Fedora, Ubuntu 16.04 and higher) use systemd to manage which services start when the system boots. Ubuntu 14.10 and below use upstart.

systemd

```
$ sudo systemctl enable docker
```

To disable this behavior, use disable instead.

```
$ sudo systemctl disable docker
```

If you need to add an HTTP Proxy, set a different directory or partition for the Docker runtime files, or make other customizations, see customize your systemd Docker daemon options.

upstart

Docker is automatically configured to start on boot using upstart. To disable this behavior, use the following command:

```
$ echo manual | sudo tee /etc/init/docker.override
```

chkconfig

```
$ sudo chkconfig docker on
```

Use a different storage engine

For information about the different storage engines, see Storage drivers. The default storage engine and the list of supported storage engines depend on your host's Linux distribution and available kernel drivers.

Troubleshooting

Kernel compatibility

Docker will not run correctly if your kernel is older than version 3.10 or if it is missing some modules. To check kernel compatibility, you can download and run the check-compatibility.sh script.

```
$ curl https://raw.githubusercontent.com/docker/docker/master/contrib/check-config.sh > check-config.sh
```

```
$ bash ./check-config.sh
```

The script will only work on Linux, not macOS.

Cannot connect to the Docker daemon

If you see an error such as the following, your Docker client may be configured to connect to a Docker daemon on a different host, and that host may not be reachable.

```
Cannot connect to the Docker daemon. Is 'docker daemon' running on this host?
```

To see which host your client is configured to connect to, check the value of the DOCKER_HOST variable in your environment.

```
$ env | grep DOCKER_HOST
```

If this command returns a value, the Docker client is set to connect to a Docker daemon running on that host. If it is unset, the Docker client is set to connect to the Docker daemon running on the local host. If it is set in error, use the following command to unset it:

```
$ unset DOCKER_HOST
```

You may need to edit your environment in files such as ~/.bashrc or ~/.profile to prevent the DOCKER_HOST variable from being set erroneously.

If DOCKER_HOST is set as intended, verify that the Docker daemon is running on the remote host and that a firewall or network outage is not preventing you from connecting.

IP forwarding problems

If you manually configure your network using systemd-network with systemd version 219 or higher, Docker containers may be unable to access your network. Beginning with systemd version 220, the forwarding setting for a given network (net.ipv4.conf.<interface>.forwarding) defaults to off. This setting prevents IP forwarding. It also conflicts with Docker's behavior of enabling the net.ipv4.conf.all.forwarding setting within containers.

To work around this on RHEL, CentOS, or Fedora, edit the <interface>.network file in /usr/lib/systemd/network/ on your Docker host (ex: /usr/lib/systemd/network/80-container-host0.network) and add the following block within the [Network] section.

```
[Network]
```

```
...
IPForward=kernel
# OR
IPForward=true
...
```

This configuration allows IP forwarding from the container as expected.

DNS resolver found in resolv.conf and containers can't use it

Linux systems which use a GUI often have a network manager running, which uses a dnsmas q instance running on a loopback address such as 127.0.0.1 or 127.0.1.1 to cache DNS requests, and adds this entry to /etc/resolv.conf. The dnsmasq service speeds up DNS look-ups and also provides DHCP services. This configuration will not work within a Docker container which has its own network namespace, because the Docker container resolves loopback addresses such as 127.0.0.1 to itself, and it is very unlikely to be running a DNS server on its own loopback address.

If Docker detects that no DNS server referenced in /etc/resolv.conf is a fully functional DNS server, the following warning occurs and Docker uses the public DNS servers provided by Google at 8.8.8.8 and 8.8.4.4 for DNS resolution.

```
WARNING: Local (127.0.0.1) DNS resolver found in resolv.conf and containers
can't use it. Using default external servers : [8.8.8.8 8.8.4.4]
```

If you see this warning, first check to see if you use dnsmas q :

```
$ psaux |grep dnsmas q
```

If your container needs to resolve hosts which are internal to your network, the public nameservers will not be adequate. You have two choices:

You can specify a DNS server for Docker to use, or

You can disable dnsmas q in NetworkManager. If you do this, NetworkManager will add your true DNS nameserver to /etc/resolv.conf, but you will lose the possible benefits of dnsmasq.

You only need to use one of these methods.

Specify DNS servers for Docker

The default location of the configuration file is /etc/docker/daemon.json. You can change the location of the configuration file using the --config-file daemon flag. The documentation below assumes the configuration file is located at / etc/docker/daemon.json.

. Create or edit the Docker daemon configuration file, which defaults to /etc/docker/daemon.json file, which controls the Docker daemon configuration.

```
bash $ sudo nano /etc/docker/daemon.json
```

Add a dns key with one or more IP addresses as values. If the file has existing contents, you only need to add or edit the dns line.

```
{
 "dns": ["8.8.8.8", "8.8.4.4"]
}
```

If your internal DNS server cannot resolve public IP addresses, include at least one DNS server which can, so that you can connect to Docker Hub and so that your containers can resolve internet domain names.

Save and close the file.

Restart the Docker daemon.

```
$ sudo service docker restart
```

Verify that Docker can resolve external IP addresses by trying to pull an image:

```
$ docker pull hello-world
```

If necessary, verify that Docker containers can resolve an internal hostname by pinging it.

```
$ docker run --rm -it alpine ping -c4 <my_internal_host>
```

```
PING google.com (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: seq=0 ttl=41 time=7.597 ms
64 bytes from 192.168.1.2: se q =1 ttl=41 time=7.635 ms
64 bytes from 192.168.1.2: se q =2 ttl=41 time=7.660 ms
64 bytes from 192.168.1.2: se q =3 ttl=41 time=7.677 ms
```

Disable dnsmasq

Ubuntu

If you prefer not to change the Docker daemon's configuration to use a specific IP address, follow these instructions to disable dnsmasq in NetworkManager.

Edit the /etc/NetworkManager/NetworkManager.conf file.

Comment out the dns=dnsmas q line by adding a # character to the beginning of the line.

```
# dns=dnsmasq
```

Save and close the file.

Restart both NetworkManager and Docker. As an alternative, you can reboot your system.

```
$ sudo restart network-manager
$ sudo restart docker
```

RHEL, CentOS, or Fedora

To disable dnsmas q on RHEL, CentOS, or Fedora:

Disable the dnsmas q service:

```
$ sudo service dnsmasq stop
```

```
$ sudosystemctl disable dnsmas q
```

Configure the DNS servers manually using the Red Hat documentation.

Allow access to the remote API through a firewall

If you run a firewall on the same host as you run Docker and you want to access the Docker Remote API from another host and remote access is enabled, you need to configure your firewall to allow incoming connections on the Docker port, which defaults to 2376 if TLS encrypted transport is enabled or 2375 otherwise.

Specific instructions for UFW

UFW (Uncomplicated Firewall) drops all forwarding traffic and all incoming traffic by default. If you want to access the Docker Remote API from another host and you have enabled remote access, you need to configure UFW to allow incoming connections on the Docker port, which defaults to 2376 if TLS encrypted transport is enabled or 2375 otherwise. By default, Docker runs without TLS enabled. If you do not use TLS, you are strongly discouraged from allowing access to the Docker Remote API from remote hosts, to prevent remote privilege-escalation attacks.

To configure UFW and allow incoming connections on the Docker port:

Verify that UFW is enabled.

```
$ sudo ufw status
```

If ufw is not enabled, the remaining steps will not be helpful.

Edit the UFW configuration file, which is usually /etc/default/ufw or /etc/sysconfig/ufw. Set the DEFAULT_FORWARD_POLICY policy to ACCEPT.

```
DEFAULT_FORWARD_POLICY="ACCEPT"
```

Save and close the file.

If you need to enable access to the Docker Remote API from external hosts and understand the security implications (see the section before this procedure), then configure UFW to allow incoming connections on the Docker port, which is 2375 if you do not use TLS, and 2376 if you do.

```
$ sudo ufw allow 2376/tcp
```

Reload UFW.

```
$ sudo ufw reload
```

Your kernel does not support cgroup swap limit capabilities

You may see messages similar to the following when working with an image:

```
WARNING: Your kernel does not support swap limit capabilities. Limitation
discarded.
```

If you don’t need these capabilities, you can ignore the warning. You can enable these capabilities in your kernel by following these instructions. Memory and swap accounting incur an overhead of about 1% of the total available memory and a 10% overall performance degradation, even if Docker is not running.

Log into Ubuntu as a user with sudo privileges.

Edit the /etc/default/grub file.

Add or edit the GRUB_CMDLINE_LINUX line to add the following two key-value pairs:

```
GRUB_CMDLINE_LINUX="cgroup_enable=memory swapaccount=1"
```

Save and close the file.

Update GRUB.

```
$ sudo update-grub
```

If your GRUB configuration file has incorrect syntax, an error will occur. In this case, repeat steps 3 and 4.

Reboot your system. Memory and swap accounting are enabled and the warning does not occur.

USEFUL APPLICATIONS FOR LINUX

50 Usefull Applications

If you’re a refugee from Windows, you may be finding the Linux world slightly confusing, wondering how you can get the all same functionality you had in Windows, but still enjoy the freedom that Linux offers. Never fear! Linux is not some scary, difficult to use monster that’s only used by hackers and programmers, it’s actually becoming more and more user friendly every day.

As such, you may be wondering what applications you should install. The following is a list of 50 useful and reasonably popular applications that many Linux users enjoy, in no particular order.

If you’re using Ubuntu or Mint, most of these applications can be found in the built-in Software Center GUI applications, or can be installed via the command line terminal. As you progress in your Linux experience, you’ll likely find yourself preferring to use the terminal, but everyone has their own preferences so do what works best for you.

```
# Terminal Command
sudo apt install [insert package(s)]
```

1. Thunderbird

From the guys that brought us Firefox, Thunderbird is Mozilla’s email client.While it technically is still maintained and various bug fixes continue to be released, there hasn’t been a major update in q uite some time. But really, how “updated” does an email client need to be? Can it send and receive email? Yes? Okay, then it’s purpose is fulfilled.

2. Geary

The default email client that comes with the GNOME 3 Desktop environment.Distros that use the pure GNOME 3 desktop environment (such as Fedora) will likely have this installed by default.

3. Evolution

Not just an email client, it also functions as a complete personal information management suite, including useful tools such as Calendar. Its functionality is very similar to Microsoft Outlook, and it can actually be connected to a Microsoft Exchange server if desired.

4. Firefox or Chrome

C'mon, you know what these are. But rather than choose one over the other, you could take the middle route and install both! Some Firefox addons are not in Chrome, and vice-versa. And maybe you want to keep certain bookmarks and addons separate between two browsers, i.e. one for work and one for play.

5. LibreOffice

The Linux replacement for Microsoft Office, this can do everything Microsoft Office can do, and you don't have to pay hundreds of dollars for it. Most distros include LibreOffice by default.

6. gscan2pdf

A simple app for scanning documents to the PDF format, which students may find useful, but is also helpful when you need to upload official documents you may have.

7. KeePass

If you're getting to the point where you have so many accounts with so many different passwords and you just can't mentally keep track of them all anymore, then KeePass will be immensely useful for you. It's a password manager that will store all of your usernames and passwords in an encrypted database. Just make sure you don't forget the KeePass password!

8. VirtualBox

If you want to experiment with different Linux distros, or different operating systems in general, you can install VirtualBox and run a virtual machine inside of your actual machine. Performance won't be q uite as good as a native OS installation, but if your system has enough horsepower you'll notice minimal performance degradation.

9. WizNote

A Linux alternative to Evernote, OneNote, and other similar note taking applications.

10. gdebi

If you're running a Debian-based system (such as Ubuntu or Mint), occasionally you won't be able to find certain software in the official repositories and you'll have to install a .deb package (or install a PPA, if applicable). gdebiallows you to execute .deb packages, which will install that particular software (you can think of .deb files as the Ubuntu/Mint/Debian e q uivalent to .exe files in Windows).

11. XDM or uGet

Download managers. I personally don't like download managers (with the exception of DownThemAll for Firefox, but that's only used in certain cases), but some people do. In my opinion, browsers' built-in download managers are sufficient. But hey, it's your system, install whatever you like.

12. UFW/GUFW

The Linux firewall. GUFW is the GUI version of UFW, which is typically run in a command line terminal. Super simple to configure and enable/disable, definitely less of a headache than Windows Firewall.

13. Gimp

Linux's answer to Photoshop. Probably the most popular image editor on Linux.

14. Pinta

If you don't need all the bells and whistles of GIMP, Pinta is a Linux alternative to MS Paint.

15. BleachBit

The Linux e q uivalent to Windows' CCleaner, with some extra features such as secure file shredding. Linux systems don't typically need much cleaning in the first place, but when they do, BleachBit is handy tool to have.

16. ScudCloud

An unofficial Linux client for the Slack messaging program (not to be confused with the Slackware Linux distro). If your organization communicates primarily on Slack, you'll want this. Slack itself also has a Linux application in Beta that's currently available for Ubuntu and Fedora.

17. Synaptic

A package manager, installed by default on many distros but not all. It provides an alternative way to install and remove packages. Many advanced users prefer it over using the included Software Center applications on many distros.

18. DropBox

Cloud storage, and they have an official native Linux client! Something that Google Drive has yet to deliver on.

19. VLC

The video player that will play anything, no q uestions asked. If you're tired of always having to find codecs on Windows, VLC solves that problem for you. It can also play audio files and can even function as a podcast manager or "podcatcher", along with several hidden features that not many people know about.

20. Unity Tweak Tool

For Ubuntu systems, some advanced configuration options aren't included in a GUI menu by default and have to be done via command line terminal. Unity Tweak Tool provides a GUI for these advanced configurations, most commonly for GTK and Icon themes. There's also a GNOME Tweak Tool for GNOME desktops.

21. Sublime

A beautiful text editor, and a favorite among many developers. It’s free to try indefinitely, $70 to buy the license, a rarity for Linux applications as the overwhelming majority are free.

22. Atom

Another beautiful text editor, without the $70 price tag

23. Notepad qq

If you’re afan of Notepad++ on Windows, you’ll be glad to know that the developer has a Linux port called notepad qq . It may not be in your distro’s official repositories, but you can always add a PPA or install the package manually.

24. Brackets

Another text editor. They all basically do the same thing, they just have different sets of bells and whistles.

25. Cheese

Linux’s webcam application.Some video chat apps may re q uire it as a dependency.

26. Gparted

The GUI version to Linux’s command line parted utility. Used for disk management (i.e. formatting and partitions).

27. CrashPlan

Backup utility for Linux. Remember, if your data isn’t backed up in at least 2 other places, it doesn’t exist.

28. Kodi

All in one media player, similar to VLC in terms of functionality. Also used to run Steam on Arch Linux.

29. Genymotion

If you develop for Android, you’ll need something to test your code on.

Genymotion is an Android emulator so that you can test your Android apps without having to actually use an Android device.

30. Tomahawk

An all-in-one music player that allows you to connect all of your various music accounts (Spotify, YouTube, Last.fm, and more) together in one seamless application.

31. GNOME System Monitor

Part of the GNOME desktop, the system monitor allows you to view information about your system in real time, such as CPU usage, RAM usage, temperatures, bandwidth, storage space, and more

32. Conky

The Linux alternative to Rainmeter on Windows. Generally used for system monitoring, conky will display lightweight “skins” on your desktop that function as persistent system monitors (whereas tools like GNOME System Monitor will open in their own windows). It’s a bit tricker to configure than Rainmeter, but you can download Conky Manager and import pre-made Conky configs. Or you can always make your own if you know how to code them.

33. Transmission

Super basic bittorrent client. Comes installed by default in Ubuntu.

34. q Bittorrent

One of many torrent clients for Linux. They all basically do the same thing, so find one that has a UI that you like.

35. Tixati

Another bittorrent client. Man, there’s a ton of these. They all basically do the same thing, download and upload torrents.

36. Steam

Never fear gamers, Steam works on Linux as well as Windows, however that doesn’t mean that every game is Linux compatible. Most (if not all) of the

Valve releases are Linux compatible, so if you're a Counter Strike or Team Fortress player, you should be able to play on Linux (assuming your graphics drivers are set up properly). Many games are Windows only though, so you might have to venture outside of your comfort zone to find some new Linux compatible games. You can actually search the Steam Store for Linux compatible games. If you have Windows games that you just can't live without, there is still hope...

37. WINE

Stands for WINE Is Not an Emulator, allows you to run Windows applications on Linux. It's not perfect, many applications and games work flawlessly, but some just won't no matter how hard you try. Your mileage may vary.

38. PlayOnLinux

A "helper" application for WINE, which can make the installation of certain games much easier.

39. VeraCrypt

The continuation of the now defunct TrueCrypt project, allows you to create encrypted volumes on your system.

40. FileZilla

Offering both a client and server version, FileZilla is easily the best FTP client available. I'm not sure who even uses FTP anymore, but it's there if you need it. Maybe you have 500GB of files that you need to send to someone?

41. Pidgin

Universal chat client, allows you to connect basically all of your various chat/IM accounts into one place (AIM, Yahoo, MSN, IRC, and more).

42. Skype

The famous video chat application, everyone knows what Skype is. However, be cautious of installing it on your system, there are some mysterious security concerns surrounding it. We suggest using whatever video chat alternatives you can find.

43. Brasero

CD/DVD burning application, comes installed by default in Ubuntu. Not many people burn CDs anymore, or even use ODDs at all for that matter, but if you still have the need to burn physical disks, Brasero will be your best bet.

44. Audacity

A basic audio editor.It's not professional studio q uality, but if you need to edit some audio for your own personal YouTube videos, Audacity is a great tool.

45. Openshot

A solid video editor for Linux.Similar in functionality to Windows Movie Maker. Like Audacity, it's not professional studio q uality, but it's more than sufficient for the everyday user that likes to make YouTube videos. I had some performance issues once I got past the 60-90 min mark, but I'm not sure if that was the fault of Openshot or if my hardware just couldn't keep up.

46. Handbrake

Desktop recording application, useful for screencasts, LetsPlays, and tutorial videos.

47. RecordMyDesktop

Same as above, desktop recording application.

48. Kazam

Another screen recording/screencasting application.

49. winFF

Video converter, super helpful if you have a video in one format but you need it to be in a different format

50. Nmap

Network mapping tool, allows you to view open ports and other various resources on your network, useful if you're security minded and you want to harden your network as much as possible.

How to Use the Linux Command Line: Basics of CLI

One shell to rule them all, one shell to find them, one shell to bring them all and in the same distro bind them.

Command line is one of the many strengths of Linux based systems. Why is it a strength? There is no one answer; there are many answers. I agree that the graphical user interface (GUI) makes it easier for a user to interact with their system and that's what new users may need to get started with Linux; that's what I needed when I was starting off with Linux back in 2005. But as I matured as a user I found CLI (command line interface) was more efficient than fiddling with the buttons of a tool.

CLI also allows users to be independent of distros. Just look at the derivates of Ubuntu, even if they use the same code-base they have different tools to do the same job. Different desktop environments on the same distro need different ways to perform the same task. A user has to un-learn and then re-learn the process of doing the same thing while they hop between distros. Furthermore if we move between Fedora, openSUSE and Arch, it becomes even more complicated.

But once you understand that in Debian-based systems apt-get or dpkg are the commands that you need to manage software, life becomes easy. Then it desn't matter whether you are on Ubuntu or Lubuntu.

When I was dependent on a GUI, I used to get worried whether that particular distro has that feature or not - it was all about certain features being exposed or hidden through the GUI. One simple example is that Gnome's Nautilus doesn't allow batch rename of files where as KDE's Dolphin does. As a result the user of x distro or DE hesitates in trying out other projects fearing they won't find the same tools. A Gnome user doesn't have to sacrifice such a useful function, thanks to the command line.

But that's not all command line does. It also saves system resources which are consumed by GUIs. So if you are on a slower system, you are better off with the command line than GUI.

People tend to think command line is difficult; it's not. It's more or less like SMSing to your PC, telling it what to do.

So without further ado let's learn some basics of command line.

Get the shell

Shell is basically a program that turns the 'text' that you type into commands/ orders for your computer to perform. As such there is a set structure of commands; different OSes may use a different structure to perform the same task.

There are many Shells available for Linux, but the most popular is Bash (Bourne-Again shell) which was written by the GNU Project. Another more modern shell with more features is 'zsh' which you can install for your distribution (we will talk about shells in a later article).

If you are using a desktop environment then you need a terminal emulator to emulate the terminal within that interface. Different distros come with their own terminal emulators: KDE comes with Konsole and Gnome comes with Gnome Terminal.

Basics Commands

When you open a terminal emulator, by default you are in the home directory of the logged in user. You will see the name of the logged in user followed by the hostname. $ means you are logged in as a regular user, whereas # means you are logged in as root.

Unless you are performing administrative tasks or working inside root directories never work as root as it will change the permissions of all directories and files you worked on, making root the user of those directories and their content.

You can list all directories and files inside the current directory by using the ls command.

```
[swapnil@swaparch ~]$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
```

Moving around

To change to any directory, use the cd command. You can also use the 'Tab' key which will auto completes the path. Use forward slash to enter directories. So if I want to change directory to 'Downloads' which is inside my home folder, we run cd and then give the path. In this case 'swapnil' is the username. You need to type your username:

```
Documents/ Downloads/
[swapnil@swaparch ~]$ cd /home/swapnil/Downloads/
[swapnil@swaparch Downloads]$
```

As you can see in the third line, 'Downloads' directory has moved inside the square brackets, which denotes that currently we are inside this directory. I can see all subdirectories and files inside Downloads directory by running the ls command.

You don't have to give the complete path if you want to move inside the sub-directory of the current directory. Let's say we want to move inside the 'Test' directory within the current 'Downloads' directory. Just type cd and the directory name, in this case it's 'Test', without any slash.

```
[swapnil@swaparch Downloads]$ cd Test
```

If you want to change to another directory just follow the same pattern: cd PATH_OF_DIRECTORY . If you want to move one step back in the directory then use cd . . /. To go back two directories use cd . . /. . /and so on.

But if you want to get out of the current directory and go back to home, simply type cd.

Seeing is believing

You don't have to change directory to see its content. You can use the ls command in following manner:

```
ls /PATH_OF_DIRECTORY
```

Example:

```
[swapnil@swaparch ~]$ ls /home/swapnil/Downloads/Test/
```

There is no place to hide

To see hidden directories and files use -a option with the ls command.

```
[swapnil@swaparch ~]$ ls -a /home/swapnil/Downloads/Test/
```

Size does matter

In order to see the size of directories and files you can use -l option with the ls command. It will also tell the permissions of the files and directories, their owners and the time/date of modification:

```
[swapnil@swaparch ~]$ ls -l /home/swapnil/Downloads/Test/
total 4
drwxr-xr-x 2 swapnil users 4096 Mar 26 11:55 Test_2
```

The command gave us the file size in a form hard to understand. If you want to get the file size in human readable format then use ls -lh command:

```
[swapnil@swaparch ~]$ ls -lh /home/swapnil/Downloads/Test/
total 4.0K
drwxr-xr-x 2 swapnil users 4.0K Mar 26 11:55 Test_2
```

If you want to get a simple list of all the directories and files inside a location, without extra info such as file size, etc., use ls -R command. This command will give a very long output (depending on how many files are there) as directory trees.

Let's create some directories

If you want to create new directories the command is mkdir. By default the directory will be created in the current directory. So give the complete path of the location where you want the directory to be created:

```
mkdir /path-of-the-parent-directory/name-of-the-new-directory
```

So if I want to create a directory 'distros' inside the 'Downloads' directory, then this is the command I will run:

```
[swapnil@swaparch ~]$ mkdir /home/swapnil/Downloads/distros
```

If you want to create a sub-directory inside a new directory then use '-p' option with 'mkdir'. I am going to create a directory called 'distro' along with a sub-directory called 'opensuse' inside it. If I run the mkdir command with '/distro/opensuse' as the path, it will throw an error that the directory 'distro' doesn't exist. That's when the option 'p' comes at play and creates all the directories in the given path:

```
mkdir -p /home/swapnil/Downloads/distros/opensuse
```

This command will create new directory 'distros' and sub-directory 'opensuse' inside it.

And now let's delete them

If you want to delete any file or directory the command is 'rm' (for files) and 'rm -r' (for directories). You need to be very careful with this command because if you fail to give the correct path of the file or directory then it will remove everything from the current directory and you may lose precious data. The command is simple:

```
rm /path-of-the-directory-or-file
```

If I want to remove the opensuse directory, the command would be:

```
rm -r /home/swapnil/Downloads/distros/opensuse/
```

However, if you want to delete all the content of a directory without deleting the directory itself use the '*' wildcard with a slash. Let's say I want to delete all the content of opensuse directory:

```
rm /home/swapnil/Downloads/distros/opensuse/*
```

If there are sub-directories inside, for example, opensuse directory then you will need that '-r' option to also delete the sub-directories:

```
rm -r /home/swapnil/Downloads/distros/opensuse/*
```

That's all for today. This chapter will make you pretty comfortable with the command line.

The Basics Of Administration And Security In Linux

Introduction to Computer Security

What is computer security?

Security is risk management. - unknown

"A computer is secure if you can depend on it and it's software to behave as you expect" - Practical UNIX and Internet Security

Security is: availability, consistency, access control, data confidentiality, authentication. - http://www.sun.com/security/overview.html

"The principal objective of computer security is to protect and assure the confidentiality, integrity, and availability of automated information systems and the data they contain." - http://csrc.nist.gov/publications/secpubs/cslaw.txt

There are numerous definitions for "computer security", and most of them are correct. Essentially computer security means enforcement of usage policies, this must be done since people and software have flaws that can result in accidents, but also because someone may want to steal your information, use your resources inappropriately or simply deny you the use of your resources.

Security Policy

A security policy is an expression of your organizations security goals. Security goals will vary greatly by organization, for example a software vendor is typically more concerned about the confidentiality and integrity of their source code anything else, while a pornographic website is probably more concerned about processing credit cards online. There is an immense amount of security technology available, from software and hardware to specific techni q ues and implementations. You cannot properly implement the technology without a solid idea of what your goals are. Do home users connecting to the office need to use VPN software? If your security policy states that all file and print transfers must either be encrypted or sent across a "trusted" network then the answer is probably yes. What authentication methods are acceptable for services that can be reached from the Internet?Do you re q uire strong passwords, tokens, biometric authentication, or some combination? The data you collect from remote sites, is it more important that this data is collected, or is it more important that this data remain secret? Data that is remote weather telemetry is probably not as sensitive as credit card numbers.

Some security policies will need to be exceptionally broad and general, for example the security policy for JANET, the academic backbone network for England states:

Monitoring and Enforcement

19. It follows from the policy of cascaded responsibility backed up by written site agreements, that there must be some method for UKERNA to enforce the possible disconnection envisaged by the AUP, and to provide full access and assistance to law enforcement agencies where necessary.

20. The JANET-CERT has therefore been given the responsibility (in conjunction with UKERNA) to

Monitor use of the network, as far as is possible while respecting privacy, either in response to information about a specific threat, or generally because of a perceived situation

Re q uire a primary site, through its nominated contact, to rectify any omission in its duty of responsibility

Where a site is unable or unwilling to co-operate, report the issue to UKERNA and initiate the procedure for achieving an emergency disconnection

Obtain evidence and pass on information as necessary in order to assist an investigation by a law enforcement agency

Provide support and co-ordination for investigations into breaches of security

On the other hand a security policiy can be fine grained:

All internal email must be encrypted with PGP or GnuPG using a key of at least 1024 bits that is signed by an authorized signing entity.

Generally speaking the more detailed and technology oriented a security policy is the harder it will be to follow and keep up to date. The actuall technical details of implementing a security policy should be seperated from it. Keeping a seperate set of best practices or an actually "implementation of security policy" document is a better idea then rolling it all into the security policy.

Acceptable Use Policy

Another component of computer security is an AUP (Acceptable Use Policy). This is a document that sates what a user of your resources may or may not do with them. Typically it is part of a contract and is signed at the time when the services are purchased. Many acceptable use policies generally forbid actions that are illegal, potentially annoying to other people (and hence likely to cause problems for the provider inthe form of complaints) or are controversial (such as pornography). Other standard clauses include "this notice may change at any time without warning" and "we can terminate your service (or fire you) at any time without warning if you violate this policy. Generally speaking the majority of AUP's are reasonable and will not be a problem for normal users. These are a good compliment to a security policy as they set out in concret terms what a user is or is not allowed to do, and as they are often part of a contract it allows a provider to enforce their security policy when a user attempts to violate it or has violated it.

Privacy Policy

Privacy policies are interesting in that they are supposed to prevent an organization 9typically a company) from violating some aspects of a user's security (specifically the confidentiality of their information). Unfortunately the majority of privacy policies contain clauses like "this policy may change at any time without warning" or are simply discarded when a company decides to profit off of a user's information (such as name, address, credit card details, purchase history, etc.).

Security is a process

You only need to make one mistake or leave one flaw available for an attacker to get in. This of course means that most sites will eventually be broken into. Witness the effects of Code Red, an Nimda, both of which were hugely succesful exploiting well known and long solved vulnerabilities in the Microsoft IIS server. Regularily apply patches (after testing them). Regularly scan your network for open ports. Regularly scan your servers with intrusion testing software such as Nessus. Audit file permissions and make sure there are no unexpected setuid or setgid binaries.

Defense in depth

All technical security measures will eventually fail or be vulnerable to an attacker. This is why you must have multiple layers of protection. Firewalls are not enough, you may accidently forget a rule or leave yourself exposed while reinitializing a ruleset, use tcp_wrappers to restrict access to services as well where possible. Restrict which users can access these services, especially if only a few users need access. Encrypt traffic where possible so that attackers cannot easily snoop usernames and passwords or hijack sessions. Since security measures will fail you also need a strong audit and logging process so that you can later find out what went wrong and how bad it is.

Technical problems

These are just a handful of thousands of specific technical problems facing security administrators.

Network Connectivity

One of the biggest security challenges is the increase in network connectivity. If you have a machine that is not connected to any other machines an attacker will generally need to gain physical access. This of course greatly narrows down the number of attackers. However with everything connected to the Internet there are over 100 million people that can potentially get into your machine.

Insecure defaults

This is one of the problems that has caused no end of security problems since day one. Vendors typically ship their operating systems with insecure defaults (i.e. finger, telnet, ftp, etc.) meaning that administrators must expend a lot of effort to close security problems before they can even start to pro-actively secure their systems and networks.

Legitimate access vs. a break in

Because you must grant legitimate users access to resources there is always the potential for attackers to gain access. Attacker can guess authentication credentials (i.e. weak passwords), steal them (password sniffing), exploit flaws in the server itself and so on.

Minimizing access and privilege

When possible restrict access. If you do not need to run fingerd turn it off and remove it. An attacker cannot exploit fingerd if it isn't present. Keep users off of servers if possible, if they need shell accounts for some reason setup a separate system and partition it off from the rest of your network. Lock down workstations where possible, set BIOS passwords, secure the boot sequence, do not give them administrative access.

Linux Installation

How you install Linux will have a big impact on how much you will be able to secure. If you install Linux from an untrusted source you could potentially end up with an installation that has backdoors or other security issues. If you install packages you do not need and forget about there is a greater chance that someone will be able to break in. As well there are security tasks that are best accomplished during installation, attempting to do them post install can be troublesome.

Verifying packages and files

Some sources of install packages and files are safer then others. Buying Linux on CD from a large vendor such as Red Hat at a large store is relatively safe and the chances of ac q uiring a trojan'ed CD set is very unlikely. Installing Linux via ftp from a remote ftp server provides an attacker many avenues to replace packages with modified versions. The good news is that most vendors have addressed this problem.

Using public/private key cryptography most vendors have a keypair that they use to sign packages, you can verify these signatures with the public key. These public keys can be downloaded online and most vendors include them on their CD's. A list of vendor key names, ID's, fingerprints and the keys themselves are available <a href="../vendor-keys/">here</a>.

You first need to download the vendors PGP or GnuPG key, for example Red Hat’s signing key has the identity of "security@redhat.com" and the fingerprint is:

CA20 8686 2BD6 9DFC 65F6 ECC4 2191 80CD DB42 A60E

Generally speaking you can download the vendor keys from their websites, and many vendors ship the key on their CD’s. To verify a directory full of RPM’s this simple script will automate the task:

```
[user@server RPMS]# for a in *.rpm
> do
> rpm -K $a >> ~/sign-log
> done
```

The file sign-log should consists of lines like:

```
ElectricFence-2.2.2-5.i386.rpm: md5 gpg OK
ImageMagick-5.2.2-5.i386.rpm: md5 gpg OK
ImageMagick-devel-5.2.2-5.i386.rpm: md5 gpg OK
Inti-0.5preview-1.i386.rpm: md5 gpg OK
```

Installation Media

You cannot secure your installation very well if an attacker manages to compromise the system before you even install it. While it is rare for an attacker to try and compromise installation media it is certainly possible. The first step is to verify the packages and installation files on your installation media, this can be done by checking GnuPG signatures on the RPM's/etc, and the other files, although finding the signatures of the other files or MD5 sums is relatively tricky it is possible. The next step is to secure the system providing the files and the path between this system and the machine you are installing the software on. This is very easy for example with a CD-ROM, and much more tricky if doing an FTP or NFS install over the Internet from a system you do not control. It is strongly suggested you use a server that you control to host installation files.

CD's

CD's are typically the easiest to install from, and relatively secure. If you purchase CD's from the vendor it is unlikely that they have been somehow compromised, all you do is simply boot from them and install the software. You can leave the machine offline for the installation typically, and if you burn the updates onto a CD you can easily update the system, again without needing to bring it online. It is strongly recommended to keep a set of vendor CD's around so that you can cleanly install critical systems as needed.

Harddrive

Installing from a local harddrive is also a relatively secure way of installing Linux. Simply partition the drive first (if you are installing Linux to it) and then copy the files onto a prepared partition, boot the system and away you go. Installing from the harddrive provides many of the benefits of installing from CD without the need to burn CD's. As well you can easily copy all the updates to the drive and then install them on the first reboot and bring the system completely up to date. Additionally with the increased availability of external harddrives utilizing parallel, FireWire and USB ports it is relatively easy to plug a harddrive into a system without needing to take it apart (assuming of course the installation floppy disk supports it).

Network (FTP / NFS)

These two methods are very convenient, all you need is a single floppy disk (typically) and a server with all the files. Simply mirror the directory structure you need (typically something like /pub/redhat/7.1/en/i386) and then make it accessible via ftp (anonymous or username/password required) or via NFS (to the IP or subnet you use for installations). The files only need to be readable, there is absolutely no need for writing to them and in fact you are probably better off removing the write bit for all to prevent "accidents". These network methods can also be combined with the automated installations such as KickStart, or you can do you own custom installation of sorts (by removing packages you do not want for example). If you choose to do network installs it is critical that the NFS/FTP server remains secure, to this end I highly recommend using a protected subnet, not attached to your main network if possible.

Automating installs

Automating installs can relieve tedium and prevent security problems. If you have to do 100 Linux installs chances are you will make error if you have to do them each manually. However if you can create an automated installation that is secured then it is much less likely that you will have problems.

Red Hat Kickstart

Red hat provides a facility for automating installs, which can be very useful. Simply put you create a text file with various specifications for the install, and point the Red Hat installer at it, then sit back and let it go. This is very useful if rolling out multiple machines, or giving users a method of recovery (assuming their data files are safe). You can get more information at: http://www.redhat.com/mirrors/LDP/HOWTO/KickStart-HOWTO.html. The configuration file can also be placed on a tftp server, in a directory named for the IP address of the client, thus using a dhcp/tftp server you can completely automate installs for machines if you know the MAC address of the client machine.

Filesystem layout and structuring

There are a number of common attacks/exploits in Linux (and UNIX) that can be reduced in risk and significance with a proper directory and partition structure. One of the most common denial of service attacks is to fill up the disk space with junk data, this can also happen unintentionally with software that experiences a problem. This is typically defended against by assigning root a set percentage of the disk space as reserve (usually 5-10%), so on a 10 gig disk users would not be able to use the last 500 megs, this would be reserved for root. This doesn't do you any good however if something running as root, that generates log files, goes nuts, or is attacked and made to generate massive log files. The next big attack that takes advantage of disk setup would be /tmp races, and core dumps, programs that create links or files improperly without checking to make sure they exist first, especially programs that run as root. An attacker can then link /tmp/foo to /etc/passwd and potentially add a user account, wipe the password database out, and so on.

Mounting options can be used to mount a partition read only, not allow execution of programs, and other useful things. You may encounter difficulties when using these options however, for example if you mount /usr as read only you will have significantly more work when upgrading system components, especially on critical servers (such as e-commerce machines) that need to be up and running, but also require critical updates. More useful options are "nosuid" (no setuid or setgid files), "noexec" (no executables) and "nodev" (no device files).

So with this in mind we have several guidelines:

Put filesystems that users can write to on separate partitions

Put filesystems with critical system components/configuration on separate partitions

Consider mounting some partitions with system binaries as read-only (this will make upgrades more difficult however), mounting /bin/, /sbin/ and /etc/

separately however will make booting the system tricky (depending on your configuration), test this before using it in a production environment.

Some notes on the various flags

noexec, if you mount /tmp noexec for example you can copy a binary in, but it will not run, however if you execute it using ld-linux.so it will work fine:

```
[seifried@stench /tmp]$ ./date
bash: ./date: Permission denied
[seifried@stench /tmp]$ /lib/ld-linux.so.2 ./date
Thu Aug 24 21:59:08 MDT 2000
[seifried@stench /tmp]$
```

Hardening your installation

So you've got a fresh install of Linux (Red Hat, Debian, whatever, please, please, DO NOT install really old versions and try to upgrade them, it's a nightmare), but chances are there is a lot of extra software installed, and packages you might want to upgrade or things you had better upgrade if you don't want the system compromised in the first 15 seconds of uptime (in the case of BIND/Sendmail/etc.). Keeping alocal copy of the updates directory for your distributions is a good idea (there is a list of errata for distributions at the end of this document), and making it available via NFS/ftp or burning it to CD is generally the q uickest way to make it available. As well there are other items you might want to upgrade, for instance imapd or bind. You will also want to remove any software you are not using, and/or replace it with more secure versions (such as replacing RSH with SSH).

Bastille Linux

If you are running Mandrake / Red Hat or a similar Linux you can use the Bastille Linux hardening script available at: http://www.bastille-linux.org/. It will disable various servers, install login banners and generally automates many of the tasks a security administrator will have to do in any event.

CONCLUSION

Linux, like most UNIX systems uses a directory structure based off of /, which directories like /home, /tmp, /usr, and so on. These directories can be placed on the same partition, or separate partitions, separating them properly can have security (and performance) benefits. There are also a number of mounting options that can be used to prevent common problems and attacks.

Thanks again for taking the time to read this book!

You should now have a good understanding of the Linux operating system, it's benefits and how to properly use it.

If you enjoyed this book, please take the time to leave me a review on Amazon. I appreciate your honest feedback, and it really helps me to continue producing high quality books.

Simply click HERE or follow this link: https://goo.gl/RtfWyz