



The Ultimate Guide to Linux for everyday people

The Ultimate Linux Newbie Guide

The official eBook from the website:

www.linuxnewbieguide.org

made with
Beacon

Table of Contents

1. Introduction

2. About the Author

3. Chapter One

4. What is Linux

5. Chapter Two

6. Why Linux - What are the Benefits?

7. Chapter Three

8. Choosing a Linux Distribution

9. Chapter Four

10. Preparing to Install Linux

11. Chapter Five

12. Installing Linux (Ubuntu)

13. Chapter Six

14. How do I get software for Linux?

15. Chapter Seven

16. How do I install software?

17. Chapter Eight

18. Using Linux Every Day

19. Quick Tips

20. How to put a Linux ISO onto a USB stick and make it bootable on a Mac

21. How to mount a USB stick as a non-root user with write permission

22. Convert images at the command line with ImageMagick

23. What is X, XFree, XOrg or X Windows?

24. How to automatically make your Windows drives become available to Linux on startup.

25. How to read and write to Windows NTFS drives as any user

26. How to use a Mac to create a Linux Live USB Stick and Boot it

27. How to install Linux on a Macintosh computer

28. SysAdmin Tips

29. Command Line Interface

30. Files, Directories and the Linux Filing System

31. How to Mount Windows or Samba Shares Permanently

32. Disable logging in with the root account

33. Quick tip: Add a user to the sudoers group

34. Adding users to groups

35. How To Set Up SSH Keys

36. SysAdmin Tips: ViM Text Editor 101 Guide

37. What is Docker (and Linux containers?)

38. running a command against every line in a textfile

39. Monitoring network bandwidth, CPU and memory effectively

40. How to tar (compress) files up, excluding certain files or directories

41. Analysing system performance with 'Top'

42. Thanks for Reading!

Introduction



Since 2001, The Ultimate Linux Newbie Guide has been helping individuals switch to the Linux Operating System. This guide can help both beginners and seasoned computer users alike learn all the important parts of choosing, using and installing Linux, a great free operating system for your computer and help you remove dependency on non-free, closed source software that is commonplace in Microsoft Windows or Mac OS.

Throughout the guide, you'll find out why Linux offers a real alternative to other operating systems, how you can install Linux on to your computer for free, and

how to get to grips with using Linux on a daily basis without any techno jargon!

On the [website](#) we also have an up to date blog with different sections including [Quick Tips](#), [Quick examples](#), a [Sys Admin](#) section and more!

I hope that the Ultimate Linux Newbie Guide helps you into a new world of freedom when using your computer and hopefully makes you smile along the way!

Handwritten signature of Alistair J. Ross.

Alistair J. Ross, January 2017.

About the Author

Since the 1990's, Alistair has been working with Linux in many capacities, either by using it himself for his career, by teaching others, and by making free software.

Alistair holds a BSC (Hons.) in Computing, as well as many certifications from various Linux based vendors such as Red Hat.

Thanks for reading this eBook version of the website, if you enjoy it, please do let me know. My contact details are below in the '*Get in touch*' paragraph.

Alistair Ross



Consultancy Services

With over fifteen years of experience in Linux, if you are in need of Open Source based support or consulting services, be it remote assistance or in-person

(Wellington, New Zealand only), then look no further than the author of this website! exposure to Linux in both large corporate environments such as [Amazon](#), [GE](#) as well as small businesses, [Alistair J. Ross](#) has been at the forefront of Linux for most of his professional life. He loves providing creative solutions and best of all, it's at a rate suitable to you or your business.

Results based outcomes with the skills to back it up!

Alistair has expert knowledge with many open source technologies including LAMP (Linux, Apache, MySQL, PHP). Web programming and databases is no problem. From CMS customisation including Wordpress, Drupal and Joomla to Kernel performance tuning, Alistair can help make your open source visions a reality. Talk to Alistair today about your open source based personal or business needs and see how he can match them to a solution that fits your needs.

Training without the Techno-babble!

Alistair prides himself in being a people person, not a techno-babble speaking geek, so if you need one on one training or class based sessions for your business, why not discuss your training needs to find the perfect programme for you.

Can you integrate with non-Open Source?

Of course! Computing only exists through the existence of multiple technologies and technology businesses. Without Microsoft, IBM or Apple - would there even be Open Source? Who knows! The fact that your business needs to operate in a world with many different technology sources means that interoperation is the reality of any outcome based business.

Get in touch!

Alistair now operates an open source consulting business, [OpenTech](#), click the below link to contact OpenTech.

[> Contact Alistair today! <](#)

Chapter One

If you are completely new to Linux, or any Operating system for that fact, this chapter covers all of the main primer aspects including:

What an Operating System is

What Linux is all about

How Linux differs from Unix

[Check out chapter one on the website](#)

What is Linux

Topics covered in this chapter:

1.1 What's An Operating system?

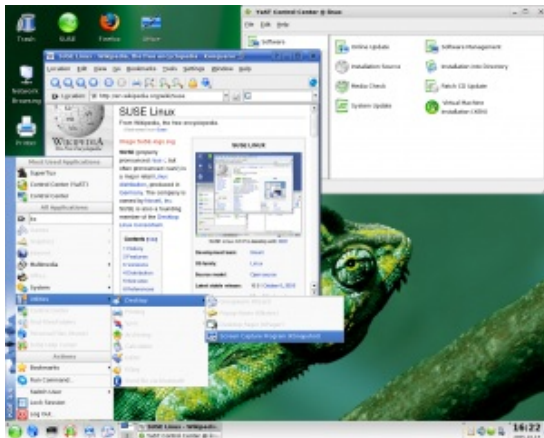
1.2 What is UNIX (and a little bit of history)

1.3 How does Linux differ from UNIX?

Linux, By Definition:

Linux is an open source UNIX-like operating system which is popular for it's robustness and availability.

The above definition is probably not going to help you much if you don't know what an Operating system is, and what this UNIX thing is, so let's start at the first major point: What is an Operating system? I promise I won't make it boring!



1.1 What's An Operating System?

Imagine you have a brand new computer. Imagine that nobody had put a disk of any kind into it, ever. That would mean that there was no software installed on the system. If you switched the computer on; It would beep a few times and then tell you that it couldn't start an operating system. The most important software to a

computer is one thing: -- the Operating System.

Without an Operating system, you couldn't surf the web, you couldn't play music, you couldn't write letters. You can't do anything.

Some of you will have heard of famous operating systems already but may not fully appreciate it. For example, Microsoft make a well known operating system called Windows, Apple make two that you may know: MAC OS X (on most Macintosh computers) and iOS (on iPhones and iPads). An operating system is the software that sits between you, the user, and the hardware inside the computer. If you click the mouse on an icon on your screen, the operating system interprets that you want to load the program that you are clicking on. For all of this to happen, The Operating system (some times referred to as the OS or O/S) must know how to use a screen (to show you what's going on), to use a mouse (so you can move it around and click with it), to use your hard disk drive (to load up the data from it). It must also need to know pretty much everything else about the hardware installed inside your computer, ie: RAM, Floppy/CD drives, keyboards, joysticks, sound controllers, graphics controllers, printers, scanners, etc.

So when you start typing a letter, for example, you have already loaded up a word processing piece of software. This software is called application software and is running 'on top' of the Operating System, but nonetheless, all of the time whilst the word processor application is running, it talks constantly to the O/S for vital information.

Okay, we've established that an O/S is necessary, but what else does an O/S do: Probably the most basic and yet essential tasks of an operating system is the job of managing our files and data. A basic O/S should be able to do the following with files and folders:

- Create them
- Move them to other directories
- Rename them
- Copy them
- Delete/remove them
- Send and receive files to/from other devices such as Printers/Scanners and your Internet connection.
- ...and a bit more.

Now you have the idea of what an Operating system is, let's find out about a specific type of operating system called UNIX...

1.2 What is UNIX? (And a little bit of history)



The operating system UNIX began life in 1969, in Bell Labs, a division of the American telephone firm, AT&T. There are now many different types of UNIX, making it one of the longest running commercial operating systems available, way longer than Microsoft Windows or Apple MacOS.

UNIX History Timeline

Linux is just one type of UNIX which is most famously known for being a free (as in free speech) derivative of UNIX. Most of UNIX's different flavours are still being updated and are still in use all over the world today. Here are just some popular manufacturers and brands of UNIX, that you may or may not have heard of before:



Sun Microsystems (now Oracle): Solaris. Developed from 1993, Solaris was a leader in the commercial UNIX world until the prevalence of open source software & Linux.



Hewlett Packard: HP/UX. HP's implementation of the UNIX standard System V, released in 1984 and is still being used today in some enterprise environments.



Berkley University: NetBSD and FreeBSD. Berkely Systems Distribution (or BSD) is the closest match to Linux in terms of a direct relationship.

With the exception of FreeBSD, there was (and still is) a pretty grand fee to own one of the above versions of UNIX. Mainly large commercial organisations and universities have traditionally used these UNIX variants, however Linux appears to be replacing traditional UNIX on a lot of corporate systems due to its proven track

record, it's growing reputation as a contender to UNIX, and it's low price tag, which can often be free.

UNIX is good because it is a true multi-tasking, multi-user operating system. This means that it can do more than one thing at a time and it can provide all it's services to lots of users at the same time. Modern day workplaces rely on servers to provide a central resource of information and connectivity to users. UNIX was also the platform that many firsts came on: The Internet, the C programming language which is the basis for most modern computer programming languages. These were all firsts that took the other operating systems like Windows and Mac OS a long time to catch up to. In fact, today, Mac OS X is built on a version of UNIX: BSD.

So, Unix is pretty clever, huh? Well, yes. It is, but Unix was also traditionally a pretty boring system that involved learning lots of commands that were tedious to learn.

Why don't we all use UNIX today if it's so good?

In 1981, a small company based in Seattle called Microsoft released an operatin **Microsoft**

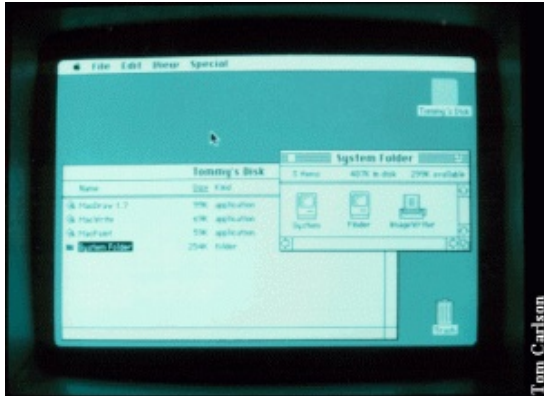
g system, which through chance (Digital Research were supposed to get the contract), were taken on by IBM to go on their new home/small office based computer: the IBM PC (or Personal Computer). This system was also not graphical. It required commands, in a similar format to UNIX or CP/M, but they were a little easier to use, at the cost of being simpler and less powerful. The main pitfall of MS-DOS - (Microsoft's PC Operating System) was, that it had no multi-user, multi-tasking or networking support as standard. By the early 1990's, this was really starting to wear on PC users. UNIX still had far more power than most operating systems of the time, it was just way too expensive, and legal issues between UNIX vendors licensing UNIX was causing headaches and therefore did not have much exposure outside of large organisations, educational establishments and government offices.



During the 80's, Apple had released another computer, which was separate from the PC, and did not run any PC software, because it relied on it's own O/S, named

MacOS. This time, Apple had decided to make an operating system that was graphical, and later, incorporated colour, pictures, icons and even sounds! Instead of typing everything into the keyboard as commands, the same actions could be made as clicks and movements with a mouse. As with all things Apple, this was revolutionary at the time and changed the face of the world of computing.

The UNIX world, still very different to the market of the PC and the Mac, not long after the mac got it's graphical operating system, began to create a graphical front-end to it's command-line world, it was called X, or 'The X Window System'.



In 1990, Microsoft eventually released Windows 3.0 (versions 1 and 2 did not sell well). Windows at the time was a 16-bit, single-tasking, single user, graphical interface built on-top of MS-DOS. UNIX still prevailed: it was multi-user, multi-tasking and it worked on 32 or 64-bit platforms.

It took until 1995, with the advent of Microsoft Windows 95 for Windows to finally go 32 bit, multi tasking, and capable of being multi-user (although not best suited: Windows NT came along shortly after, to do that job).

1.3 How does Linux differ from UNIX?

During the time from 1991 to 1995, many computing or engineering students were accustomed to the power of UNIX and X, at university. In Uni, most students had wonderful new things like E-Mail, The Internet and more. At home, they would have to make do with their 16 bit computers, waiting for all these powers to come to their homes one day.

Enter: Linus Torvalds

Linus was, in 1991, a student in Finland studying Computer programming at The University of Helsinki. Linus used UNIX at University on a daily basis. He got bored of his 386 PC running MS-DOS, and decided to start his own kernel, which is the

name for the code at the heart of every operating system that talks to the hardware directly. He wanted to distribute the software freely, because it was a hobby, not a commercial product, and also to see what others thought of it. He finished the first Linux kernel in late 1991. Not only had he made a 32 bit kernel, in which programs could be run he made it do quite a lot to make it look and feel like Unix, but he didn't have any software to run on it.

Here is Linus' very first post to the Internet in 1991, about the creation of what would become Linux:

*Hello everybody out there using minix -
I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready.*

Luckily, an ex-student in the USA, by the name of Richard Stallman had created a team of programmers devoted to free software, he called this the Free Software Foundation, who believed in making software free to distribute, and free to obtain the source code along with it so that others could make improvements to the software through the Internet. The GNU GPL (General Public License) that the Free Software Foundation made, also stated that the authors of the software could charge for the software, as long as they are willing for it to be freely distributed. By creating community-based software, that has open standards and is subject to peer review, the quality of the software would be good. The opportunities for profit would come from different avenues such as support and consultancies.

Stallman had been busy making a whole suite of software, for example: a text editor called emacs, and bash (the Bourne Again Shell) which is a command line interface based upon the original Bourne Shell that comes with the BSD variant of UNIX. The FSF's software was entirely based upon the UNIX software suite, and essentially improved on it. In 1991, the only thing that the FSF were missing to make it a fully fledged operating system was The Kernel.

Linus altered the code to work on his platform so that the FSF's code would work with his new kernel, that he ended up calling Linux.



Linux is pronounced 'Lih-nucks' not 'Ly-nucks'

Here's Linus Torvals pronouncing it!



Now, over twenty years on from the original post on the Internet, Linus is still working on Linux, but it's now an effort which is collaboratively worked upon by millions of other individuals, corporations and organisations around the world. You probably don't know it, but Linux is everywhere today. It's in your TV and media-centres, it's the basis to that Android mobile phone you probably have, nine out of ten websites run Linux on their servers (think Google, Amazon and Facebook to name but a few), Linux might even be in a new dishwasher or fridge-freezer!

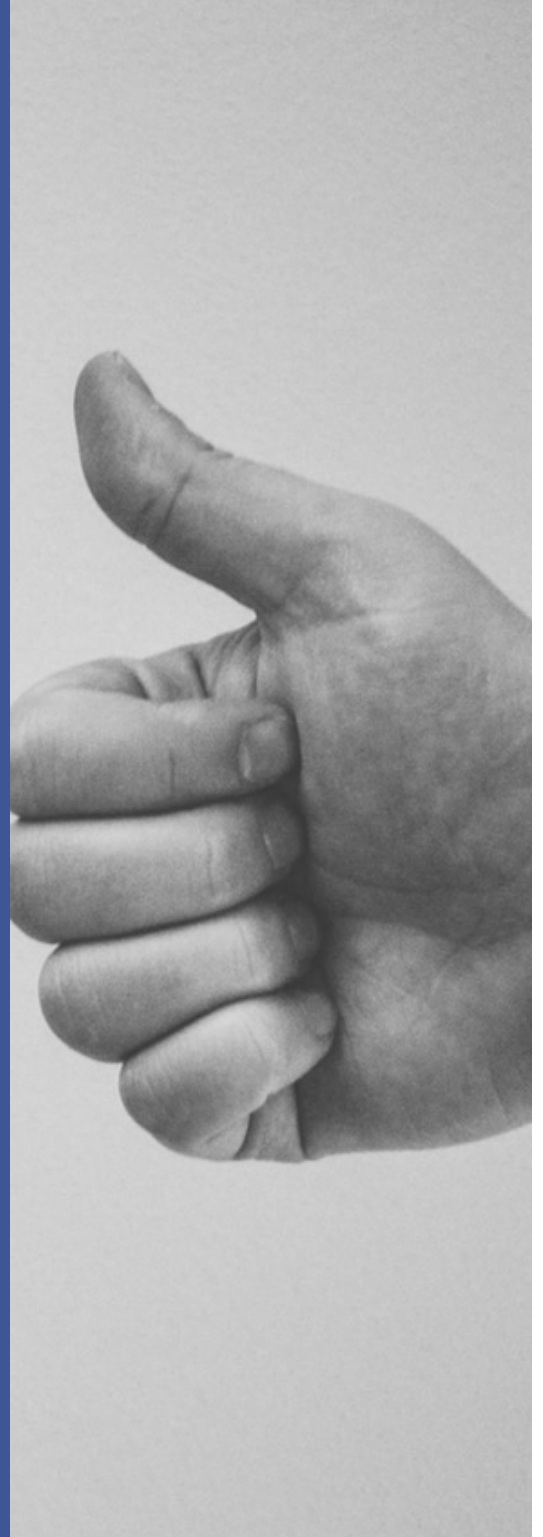
Now, visit [Chapter Two](#) to see why Linux may be the best thing you ever did with your computer!

Chapter Two

Why Linux? What are the benefits?

Chapter two answers the burning question that you really need to know before going out and just installing a new operating system: Why do it and what does Linux actually offer me?

[Check out Chapter 2 on the website](#)



Why Linux - What are the Benefits?

So what does Linux actually offer me then?

So, you now know that Linux is a Unix-like operating system, and you know what all that means now. However, that doesn't really tell you why you would prefer to use Linux, instead of Windows or Mac OS on your computer.

Linux is far more than a Unix-like operating system and is pretty unique because of its unique Licensing system, enter 'Open Source':

Linux is truly an Open Source Operating System:

- The open source GNU General Public License, that Linux uses means that you can obtain the software free of charge, and you can obtain the source code to the software and make changes to it if you want. You can then re-distribute it if you like, provided you supply the source code with your changes too.
- Open standards provoke less buggy software because it is worked on by a global team of developers from many far reaching backgrounds.
- Open standards also mean that compatibility across any other open platform. For example, you can be sure that an open source OGG audio file will play on any OGG player in exactly the same way, because the open standard applies throughout.
- Open Source software means no vendor lock-in.
- Trustworthy computing, as the source code for all software is distributed for free, often with the applications you obtain.
- No chance of Linux as a whole going out of business, as it is not owned by any one company.
- Did we mention that Open Source software is generally (but not always) FREE in cost?

This means that software can be of high quality for everyone, and money can be made out of support, distribution, training or working with OSS.

It's a new and revolutionary way to do business, but already, huge names like IBM, HP, Novell, Sun, Intel and even non IT firms such as Boeing, Glaxo Smithkline and thousands more are all using it and putting work back into it.

It's a reliable platform for any sort of mission-critical work:

- Compared to other platforms, Linux deals very well with rogue software and as a result, you are very unlikely to have the whole system crash on you.
- There at present no mainstream viruses for Linux. Viruses would need to change in their nature drastically for them to be plausible in the Linux arena.
- It's fast and works on computers that are pretty old, recycle that old PC today!
- Say goodbye to Malware, Spyware, Adware etc.
- Security is built-in by default, not an add on.
- Linux is home to some of the best new software, and best of all, most of it is free, have a look at just a few of these great titles:



OpenOffice.org/LibreOffice

A fully featured Office suite including a Word Processor, Spreadsheet, Drawing Package, Database and Presentation suite. Compatible with Microsoft Office.



Mozilla Firefox (and Google Chrome)

The browser that reloaded the web - all thanks to Open Source!

The GIMP



(GNU Image Manipulation Project)

An excellent open source image editor, similar to Photoshop by Adobe. Used to edit the images on this website!



VLC Media Player (VideoLAN)

A great media player, play your DVDs/VCDs/DiVX's on any computer, even stream them to another computer connected to a network! VLC is one of the most popular media players available today, and it's also available for Mac OS X and Windows users, too!



Evolution

Evolution is a full-featured Groupware client which includes E-Mail, Calendar, Tasks, Address Books and the ability to connect to a Microsoft Exchange mail server.

...and thousands more titles, have a look at these links for more great examples:

- InfoWorld's BOSSie Awards:
<http://www.infoworld.com/article/3122000/open-source-tools/bossie-awards-2016-the-best-open-source-applications.html>
- Datamations's Open Source Ultimate Software List:
<http://www.datamation.com/open-source/open-source-software-list-ultimate-list-1.html>

How can this make any business sense, does anyone make any money?

You might think that as Linux and the associated open source titles that go with it are free in cost and also free 'as in speech' that this means that there is no money to be had from Linux. Indeed, many companies originally thought that Linux was nothing more than a hobby or a geeks plaything, but this perception has diminished over the years, and with Linux going strong since 1991, it's here to stay. Here are a few reasons why Linux helps businesses and can generate profit over the traditional software business model:

- Linux is one of the most popular and reliable platforms on earth today, it is this basis that has let many companies such as Google and Amazon build from that foundation and leverage it to make solid profit. Every Android mobile phone or tablet uses Linux, all of Amazon's websites use Linux, every Google search is powered by Linux and every Tweet uses Linux at the operating system level.
- Many businesses choose to purchase support contracts to obtain help with their Linux systems, just as they would with a commercial based platform.
- Companies like Intel, who invest heavily into Linux can see the return on their investment rapidly because Linux has the flexibility to allow their newest technologies such as new processors to work straight away via the Open Code/Source model. Typically, Intel would have to wait many months or years to see support fully phased into releases of Windows or MacOS for their latest products.
- In countries like China, up to 70% of all the computers that ship now ship with Linux on them. The vast majority of these computers retain their installation of Linux because the Linux installation allows the user to do what they want to do with their computer, it means no levy for the manufacturers to pay to companies like Microsoft for copies of Windows and therefore cost savings to the customer. This in return has shown a rise in computer purchases and sales at a better margin of profit.
- Many companies are switching to Linux for server hosting because it will outperform its Windows counterparts and will do it at a near zero-cost. This provides a hefty return on investment to the company, and also in turn to their customers. This can be stacked up with other cost-saving methods such as virtualisation, which means that you can have multiple Linux servers all running on one physical server. This reduces costs in the server room on cooling, power and hardware. For many companies across the globe, Linux is therefore a no-brainer.
- Finally, Over time, many companies, universities and hobbyists find themselves naturally giving back some things to the Linux community because it helped them with a certain task. This continued cycle of improvement and collaboration spurs this on. You only have to look at open source projects like Wikipedia to see that this cycle works very well. Nobody gets paid to make Wikipedia the best source of information anywhere in the world, but yet people add to it none the less. Linux works in the same way.

Now, visit [Chapter Three](#) to decide on which flavour of Linux you want!

Or, if you need more convincing, why not take a sneaky peek at [Chapter 8](#), where we detail how Linux helps out with your life on the desktop every day.

Chapter Three

Choosing a 'distribution'

If you have not yet Installed Linux on your computer, you might want to have a look at this chapter for information on choosing a distribution (a flavour of Linux) that suits you. There are literally hundreds, if not thousands of different flavours out there, however a few stand out from the crowd. This chapter discusses those distributions and gives you a few handy links to places where you can find out information about other distributions.

[Check out Chapter 3 on the website](#)



Choosing a Linux Distribution



With so many Linux distributions, it can be hard to choose, so we help you select one that's right for you.

As described in the first chapter, we discovered that Linux was a flavour of the UNIX family of operating systems. This chapter talks about what types of Linux are available in the market today. These flavours are called distributions and all have their own merits and disadvantages. We will cover the most popular distributions in

this chapter.

3.1 What Exactly is a Linux Distribution?

If you ever watch the IT press, you will have probably heard of company names such as Red Hat, Mandriva, Canonical (Ubuntu) and Novell (SuSE). There are literally thousands of other smaller companies and organisations that also make Linux distributions. Examples of which can be seen on websites like [distrowatch](#).

These are all companies or organisations that have created their own 'distributions' or flavours of Linux.

In any distribution, the fundamentals stay the same:

- There is always a Linux Kernel (the core component of the Linux operating system)
- The default GNU software (tools like ls, rm, etc)
- General software to be expected of a Linux distribution (text editors, etc)

What differs from distribution to distribution is usually:

- Installation Software (for installing software, or the operating system)

- General software: (Office Apps, Prog. Languages, Games, Web Software etc)
- Documentation and Manuals (Quality of, Lack of, Quantity of)
- Cost - whether you pay nothing, a little, or a lot for a distribution depends on what you need from it and the business model the distributor works to.
- Quality of software (buggy or not buggy software, latest versions of software)
- Whether it is up to date or not
- Whether the distributor offers a good channel of support or not
- How easy it is to use overall.

As you can see, whatever distribution you choose. You get Linux, whatever distribution you choose. You may get a better range of options with distribution X over distribution Y, The choice is for you to decide, and because of Linux's excellent Copying/Licensing properties, you can often download a distribution from the net, or have a copy made for free by a friend without having to part with any cash.

Bundled Software with your Linux distribution

When you buy a mac, or a Windows PC you will find you get a nominal amount of software with it, made (usually) by either Microsoft or Apple. If you buy the PC from a vendor such as HP, you might get some extra tools as well like Multimedia software, but what you will undoubtedly notice is that you have to buy or download most of the software you want to use on your system after you get it out of the box. With a Linux distribution, you generally find that it comes with a plethora of software already pre-installed, from many different companies or individuals. Some large software like office suites (OpenOffice.org) to smaller but powerful tools like DVD/movie software such as VLC. It is likely that for a good number of people using Linux on the desktop, you already get all the software you want without ever needing to install anything more! For more information on software within Linux, see chapter 10 or view some of our video tutorials.

A little bit about 'Live' distributions

In more recent Linux history, so called 'Live' distributions have become very popular, because it let's you try out a Linux distribution without even installing it onto your hard disk. This is great because you don't have all the hassle with repartitioning the disk (see chapters four and five) or installing over your Windows/Mac OS software. You can simply drop the CD for a Live Distro' into your CD Drive and start up your computer from that. You usually get most of the main functionality of the distribution so you can really evaluate if the distribution is for you before you choose to install it for real. In fact there are web sites out there

that allow you to boot Linux straight off a USB pen drive such as pendrivelinux.com. Canonical's Ubuntu Linux has a great installer that has an option to install Ubuntu on the same disk partition as your Windows XP or Vista setup using an installer called the 'wubi installer'. It runs a tiny bit slower than normal Linux, but it's a very easy way to install Linux alongside Windows if you want to test the waters first.

Linux may be free, but can't you also buy Linux? Why would I do that if I can get it for free?

Buying Linux often provides benefits that downloaded versions do not provide, such as:

- Physical manuals (SuSE & Red Hat Enterprise Linux are particularly good) to help you out when you need a 'covers-all' reference.
- Vendor support for a particular period of time
- Distributions like Red-Hat Enterprise give corporations a guaranteed Service Level Response
- Sometimes you may get more software than with other distributions (eg extra DVDs instead of downloads).
- Commercial software titles can be included (as it is non-free), these can include copyrighted or patented technologies such as DVD and MP3 players, as well as commercial software like Adobe Flash Player and so forth.

3.2 What Linux Distribution should I choose?

Choosing a Linux distribution is a personal thing. It greatly depends on what you want to do with it.

If you like the look of a Linux Distribution and want more information, click on the appropriate distribution logo below to visit the distributors web site. This is just a very short collection of some of the more popular distributions out there. For more in-depth information on the differences between each distro, we would recommend visiting distrowatch.com

Here is an example of just some of the more popular Linux vendors today:
Vendor Logo User Level Good Points Bad Points

Ubuntu and Linux Mint



Suitable for: Beginner to Advanced/Server

Ubuntu is currently the most popular of the Linux Distributions. It is built on a Debian core, but has a more regular release cycle, is more polished, is easy to use and has major financial backing. It is a completely free distro, therefore copyrighted materials such as DVD playing software do not come as standard with Ubuntu, you must download and install it separately, but can be done easily. If you don't like the look and feel of the latest Ubuntu desktop (called Unity), [Linux Mint](#) is based on Ubuntu, is made for beginners and still offers a GNOME or KDE version.

Red Hat/CentOS/Fedora



Suitable for: Beginner to Advanced/Server

Used to be very popular, easy to use, good installer. Has some annoying quirks, RPM software packaging can suffer from dependency problems, even with YUM system. RHEL (Red Hat Enterprise Linux) is the non-free Enterprise version offering of this distribution, it comes with full telephone based support and is backed by rigorous testing. CentOS is the free version which is derived from RHEL but usually trails behind it and of course does not come with enterprise support, then there is Fedora Core. Fedora is the bleeding-edge fork of Red Hat which has all the latest bells and whistles but as it is bleeding-edge, it can also suffer from less stability than their enterprise-grade counterparts.

SuSE Linux



Suitable for: Beginner to Advanced

SuSE was once an independent German Linux distribution, which later was purchased by Novell, who later sold it to VMWare. It's now an excellent all-rounder which is geared up for the Enterprise. good manuals & docs, masses of great software, brilliant support. Enterprise version great for corporate use with business support and has partnered with companies such as SAP (and of course VMWare). Software Installer still relies on RPM system from RedHat which can suffer from dependency problems however this is mainly a thing of the past.

Slackware and Arch Linux



Suitable for: Advanced to Server Users

Slackware was probably the first linux distribution. Targeted at geeks who like to

tweak or for the server market looking to get every little ounce of server performance. Quite hard to install and use, Uses .tar.gz packages rather than more popular .deb or .rpm systems. If you fall into the more advanced camp, but don't like the sound of compiling everything, perhaps Arch is for you, as it still offers similar levels of customisation as Slackware.

Debian



Suitable for: Intermediate to Advanced Users

Very established Linux distro. DEB packages combined with apt-get system solve the tedium of the RPM software packaging in Redhat/Suse/Mandriva. Traditionally known for being further behind than some other distros, but rock solid. Is now the basis for many modern, easier to use distributions such as Ubuntu and Linux Mint.

Note: If a distribution is at release 10 (ie: Slackware), but another distribution is only at 4.1 (ie: Debian), this does not indicate that Debian is an old version of Linux. The release numbers are only an indicator of how many releases that particular vendor has made. For example it is quite likely that Debian 4.1 and Slackware 10 share the same major kernel version and many similar software titles.

Check out The [DistroWatch Linux distribution popularity rank](#) (Page Hit Ranking) for a good idea on what's hot in the world of Linux Distributions right now, it's updated daily!

If you are ready to start preparing to install Linux on your system, then move forward to [Chapter 4](#)

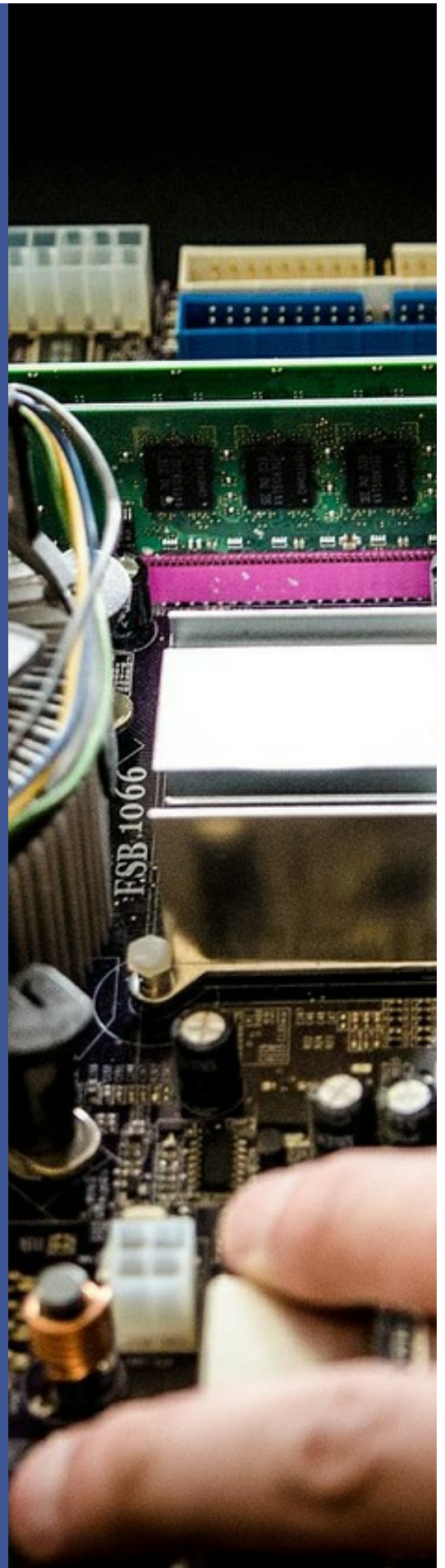
Chapter Four

Preparing to install Linux.

This chapter deals with ascertaining what you need to do to prepare for installation of Linux, including ensuring all your hardware will work with Linux.

There is also information about how different versions of Linux work with regards to software/package management.

Check out Chapter 4 on the website



Preparing to Install Linux

4.1 Different types of Distribution and installing software with them.

As described in the previous chapter, there are lots of different types of Linux distributions floating around, and yes, this generally means that they all have different Installers to put them onto your PC, Mac or Alpha based computer. This chapter focuses on installing a Debian based distribution, but gives a short insight to other distributions as well.



Debian is one of the oldest distributions out there. Slackware and Red Hat are pretty much the only other two that come close in age, Debian has lasted the test of time, and does not look like it's going anywhere soon. It's a free distribution (which will not change, because of it's license) and it has thousands of developers world wide.

A major difference between Debian-based distributions (such as Ubuntu and Linspire) is the fact that they use the DEB package management system to install software.

Installing software via binary packages, or if you need the source code, via source packages, is very convenient for most Linux users, because it means that you don't have to compile the source code of an application to get it working. In most cases, you can simply click on a package to install one via a tool in a GUI, and the software will be installed. Here we will discuss two major package systems, however, there are others, such as Slackware's .tar.gz based system, and BSD's

ports system.

When the .DEB package format is combined with software such as APT or Synaptic, the .DEB system works very well at resolving things called dependencies (software that needs other software, in other to work).



The Red Hat Package Manager (RPM) Logo

Red Hat based distributions such as Fedora, SuSE and others use the RPM Package Manager, previously known as the Red Hat Package Manager. Packages have a .RPM extension (for example gimp-2.05.i386.rpm) are packaged binary applications (sometimes they package source code as well).

For software that is not bundled with your Distribution, you should read Chapter 9, "How Do I Install Software?".

Distributions like Red Hat, Fedora and SuSE seem to provide less bundled software packages for their distributions than the DEB based ones. This is mainly due to something called the Debian Universe which is a large repository of software which is available in internet repositories. The universe is not supported and as-such should be treated as software in the 'wild', but it is freely obtainable through the same software installation tool (such as Synaptic) as the Distributors software. RPM based users often have to visit third party websites such as freecode.com to download packages, as well as any dependent packages (dependencies). Both the use of the Universe, and third-party websites can have issues, because they contain software that is not guaranteed to work with your distribution, and may cause unexpected results - so be careful if you download software from other sources!

Both RPM and DEB packages are very widely used in the Linux arena, most of which can be installed simply by using software, like the 'Add Applications' menu in Ubuntu.

4.2 What sort of computer will I need for Linux?

This question has a lot of answers. The bottom line is:

Depending upon what you want to do with Linux, the system requirements can range from an old Intel 386 to a state of the art PC. You can even run Linux on

some stranger hardware including Macintoshes, ARM based machines and more.

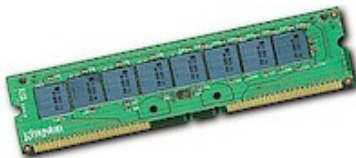
This section of the chapter will go through all the major parts of hardware attached to a typical PC and detail what is expected to run a typical modern Linux desktop, starting first, with the CPU.

CPU (Central Processor Unit)



Linux was originally devised on an Intel 386 back in the early 90's, this however does not necessarily mean that Linux works better on a PC than a Macintosh computer (PPC or Intel). Today's Linux desktop is most popular on Intel, PPC (G3-G5) and AMD processors, therefore, most of the common software is actively developed for these platforms. If you have another platform such as a Sparc, Amiga, Atari or ARM based processor, Linux will no doubt be different in that many software titles may not exist for that given platform, or software is older than that of the most popular platforms, but it is still possible to run Linux on them.

RAM (Memory)



Most Modern day Linux distributions will require a minimum of around 1GB to use it to a reasonable degree, but if you wish to use Linux for non-graphical based uses, such as web page hosting, or a firewall, you can run a basic installation of Linux from almost nothing. Some of the most basic installations will run on 8 MB (yes megabytes, not gigs!). If you're going to be serious about Linux, and want optimal performance, then as with any software, the more RAM you have for it, the better it runs. Ideally, if you reckon you're going to be a home user, at least 1GB RAM is required. If you want to do demanding stuff like perform movie editing, edit artwork or edit lots of audio, then we're probably talking about 2GB+. Server users who want to serve up hundreds of websites may want 4GB, 8GB or even more, but again, if you want to make a small server with only a website or two and a low number of users, then you can get away with 1GB or less.

In summary, If you have the RAM, Linux will use it, and it will be used well, thanks

to the superb memory and process management within the Linux kernel a modern-day 64-bit version of Linux will support up to 64 TB (terabytes) of RAM.

Hard Disk Drive (HDD) & Partitioning your disk for Linux



As with all things Linux, it's possible to do it in the smallest of setups. Using distributions such as ZipSlack or Puppy Linux, you can achieve a fully working Linux setup in a few hundred megabytes. However, if you want to install a standard workstation installation of any up-to-date distribution, you will probably want at least 20GB (gigabytes) free hard disk space. If you are going for the plunge and will convert your entire system over to Linux, then the more the better - 100GB+ in order to store all of your stuff: Apps, MP3s, Movies, Documents, emails etc and over time, it uses up quite a lot of drive space.

Modern Linux distributions easily support new drive technologies such as software RAID and SATA, so you will have no worries about your latest technology drives. If you use more high end storage equipment such as iSCSI or fibre channel disk arrays, distributions like Ubuntu Server edition support these technologies right out of the box.

As you are just starting out, you may find it easiest to purchase a new hard drive to install Linux on or why not recycle an old hard drive if you have one spare!

The reason for using a separate drive is because you are likely to be using another Operating System already such as Microsoft Windows or Mac OS. If you wish to use both Linux and Windows/Mac OS (so you can see if Linux is for you), then the easiest way to set it all up is if you have another drive to put Linux onto. You won't have to mess around with resizing partitions and the like:

(Re)-Partitioning

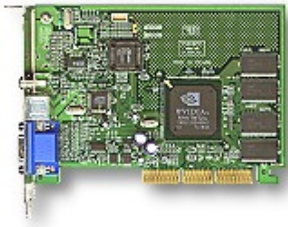
Typically, Windows/Mac OS will allocate 100% of a hard drive to its own use, meaning there is no space left for Linux. If you don't want to buy a new hard drive for Linux, then you will somehow have to re-allocate some of the unused (free) space on your Windows/Mac OS drive for Linux.

Thankfully, recent versions of Ubuntu and some other Linux distributions now make it a snap to re-partition your disk. They work by utilising the free space that you have in your Windows drive (say your C: Drive) and creating a partition out of some or all of that free space for Linux. If you don't want to use Ubuntu or similar,

you can also use something like the freely available [Parted Magic](#) or commercially available [Acronis Disk Director](#) (for Windows) which splits your disk into partitions as well as resizes existing partitions can make it a snap to have Windows and Linux exist side-by-side on the same disk. It is very easy to use for the beginner.

Although the process of re-partitioning and dual-booting your PC with Linux and Windows is far easier than it used to be, to a computer novice it can still appear to be a daunting task. Don't worry though, I've got a video demonstration on how to do just that in [Chapter 5](#).

Video Card (Graphics Adaptor)



Any bog standard graphics adaptor will do for linux. Optimally you will want to have an SVGA adaptor in your PC that has enough RAM to support resolutions of at least 1024x768. Graphics Accelerator cards of many types are supported by today's modern distributions for even faster graphics. If you're looking for really good graphics performance under Linux, the NVidia range are an excellent choice, because they are well supported under Linux by Nvidia. ATI cards are also popular, however their driver support for Linux does not appear to be as good as NVidia's, which seems to be an ongoing issue with ATI. If you don't know what card you have in your machine, visit your device manager in Windows, or System Preferences in Mac OS. Integrated graphics chipsets such as the Intel i Series or Cirrus Logic chipsets generally work ok, however if you need 3D graphics performance, as with Windows or Mac OS, you are best using a 3D Accelerated graphics card from the likes of Nvidia or ATI.

Using Wireless & Wired Network Adapters and (Broadband) Modems with Linux



Almost every wired network card available should be quite happy with Linux. Modern PCI or integrated based options such as those manufactured by 3com,

Intel and Realtek range will automatically plug and play.

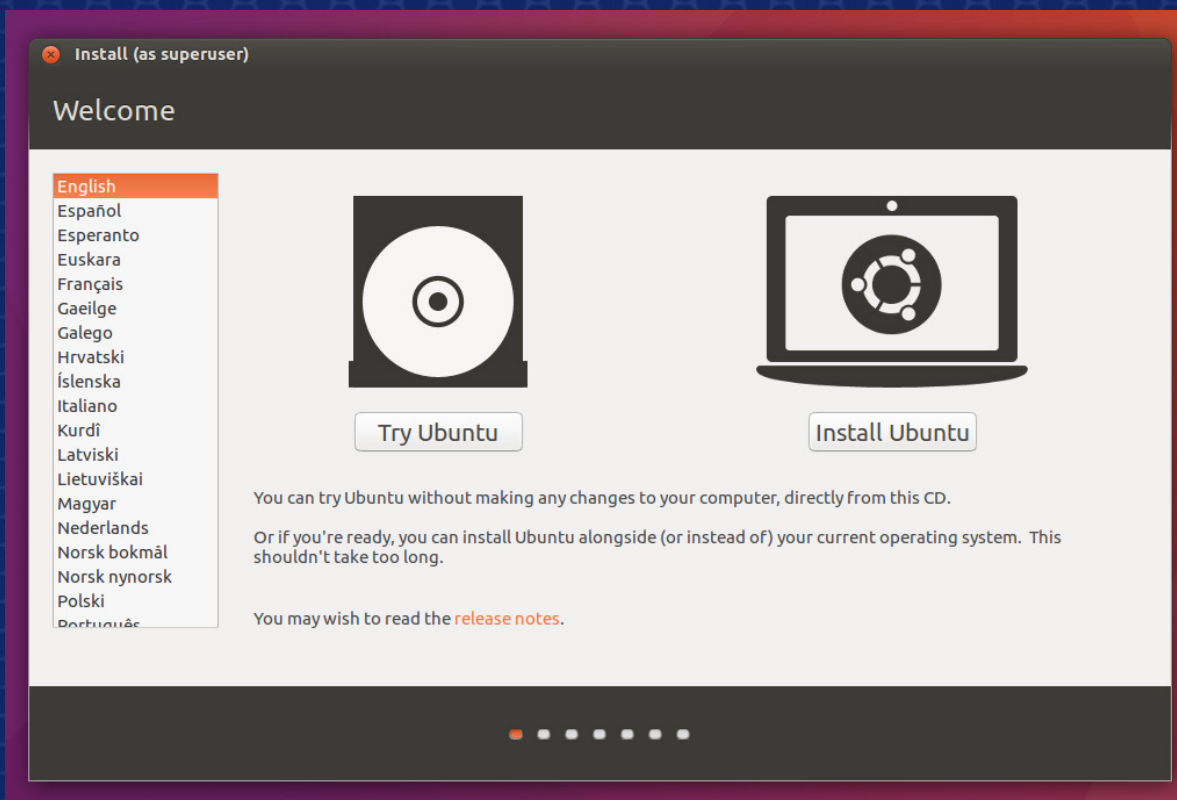
Wireless card support in Linux is generally good. Standard desktop PCI based WiFi cards will work out of the box without any need to install a driver. However, some vendors have made cheaper soft-pci, mini-pci 'wintel' based cards (a lot of which are found in laptops), these do not work as well as their larger counterparts under Linux because the vendors do not wish to provide drivers or any information for Linux developers to work with. This can usually be resolved by loading the Windows driver inside Linux, using a tool called [ndiswrapper](#) (see [this wikipedia link for further information](#)).

Cards known to work out of the box include the Orinoco chipset, Intersil Prism/Prism II and Cisco Aironet based cards. For more information on Wireless compatibility under Linux, see [the Linux Wireless LAN](#) wiki.

ADSL & Cable modems are usually one of two breeds, either they either plug into the USB port of your computer directly or they are fully blown Ethernet routers, today these mostly contain WiFi radio as well. Thankfully, most ISPs are now providing 'proper' Ethernet based routers which simply plug and play with Linux either over WiFi or via an Ethernet cable. If you do have a USB modem from your ISP, consider shelling out for a proper router as the USB modems support under Linux is somewhat hit-or-miss and you will often find that performance from a USB modem is less than you would get from a router (regardless of whether you are using Windows, Linux or a Mac). Good examples of external routers are Belkin, Netgear, Linksys and Draytek.

Now, armed with all the information you can get about the hardware in your computer, it's time to get Linux installed! [Click here to visit Chapter 5 to do just that!...](#)

Chapter Five



Installing Linux (Ubuntu).

This chapter gives a full step-by-step guide to installing a popular Linux distribution.

It shows full details and of each stage and gives full details of all the choices along the way, including information about partitioning your hard disk drive.

On the website there is also a video of an Ubuntu installation which you can watch as you go along:

Visit Chapter 5 on the website

Installing Linux (Ubuntu)

5.1 Installing a Linux Distribution: Ubuntu

Although this example shows the installation of the Ubuntu Linux distribution, installing most other Linux distributions is a similar process. I have chosen Ubuntu as it is a friendly, free, highly compatible distribution of Linux and at the point of writing, it has been the most popular Linux distribution for quite some time.

Please also note that this tutorial details the installation of Ubuntu Linux on a PC, if you have a Mac, the instructions are similar, but not the same. In particular, the tools you will use for partitioning your hard drive may be different.

5.2 Planning the Task ahead

If you remember back in [Chapter 4](#) we discussed hard drives and partitioning. If you didn't read that part, to have a fuller understanding about what we're about to do with your hard disk, skip back and read it now.

WORDS OF CAUTION!

At this early point in the process of installation, be aware that you will be working with your hard disk in order to install Linux. You **MUST** make a backup of that data before starting. You do all of this at your own risk!

The main step with a Linux installation is to virtually 'slice' up your hard disk into partitions in order to put Linux onto it. This process is called partitioning. You will not have to perform this step if you have chosen to use an additional new or recycled hard drive to install linux onto. Also, if you are brave and wish to simply delete the operating system (Windows/Mac OS) clean off your computer, then this step is also not necessary, otherwise, proceed forward!.

How to partition the disk

Since as far back as 2007, it has been possible to re-partition your disk drive during

the installation process, making it easy to do, so this guide has been updated to follow this process. Read on and watch the below videos to learn how this is done.

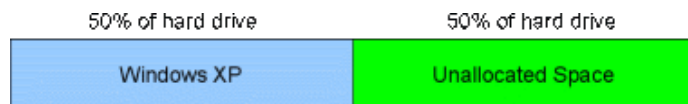


Tux tip: If you still want to partition your hard drive before installing Linux, for example if you want completely granular control of how you split up your disk partitions, even after you have installed Linux, then you can use a tool like [GParted](#), or [these others](#) to partition your disk so that you can create some free space to put Linux onto.

The video below in section 5.4 will demonstrate how to partition your disk step-by-step before we go on to the video, here's some information about partitioning you will want to know:

If you are resizing your Windows partition to accommodate the installation of Linux, try and devote as much space to Linux as you can manage, if for example you have 40GB unused/free space on an 120GB drive, resize your windows partition down from 120GB to 90GB, leaving 30GB for linux and 10GB room to spare for windows. This way you probably won't have too much concern about free disk space in the future.

You can split it any way you like, here is an example of how your hard disk, if drawn as a sideways graph would look if you split it 50/50. The Unallocated space would be later formatted under the 'ext4' file system by Linux during installation:



A graph demonstrating a 50/50 split of a hard drive. Windows uses the NTFS partition type (on the left) and the Unallocated space will be utilised by the EXT4 (Linux) file system during the installation process.

5.3 Downloading and starting the Ubuntu installation

In case you didn't read over the previous chapter, firstly, make sure your PC is up to the job of working with Ubuntu.

You will need:

- A PC with at least 2GB RAM, 40 GB free space on your C: drive or target install drive.
- A USB stick which is empty and is at least 2GB in size (or a DVD-R)
- An Internet connection to download the ISO image (approximately 1GB in size)

If that all checks out, then you are going to need a copy of Ubuntu, so head on over to their [website \(https://www.ubuntu.com/download/desktop\)](https://www.ubuntu.com/download/desktop) and download a copy.

When it's downloaded, you'll end up with a .iso file. This file type is what is known as a *disk image*.

Copying the 'ISO' image to the DVD or CD.

Copying the ISO image to a DVD is not the same thing as just dragging and dropping the file from your Downloads folder and copying it onto a blank DVD or USB stick, you need to burn or unpack the disk image onto the media first.

Unpacking the ISO file to a USB stick

If you prefer to use a DVD-R, rather than a USB stick, then skip this section and see below.

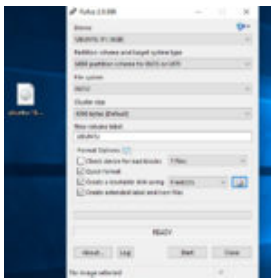
There are a few ways to put an ISO image onto a USB stick these days, and it's a lot easier than it used to be. As most of us have USB sticks lying around the house, this is now the most common way to install a Linux distribution.

For Windows users, the easiest way right now seems to be by using the free tool, Rufus:

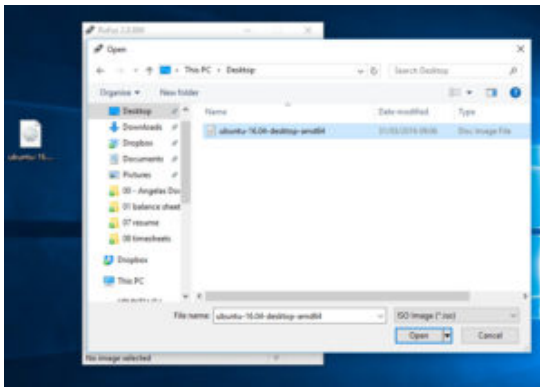
[Download it here \(https://rufus.akeo.ie/\)](https://rufus.akeo.ie/).

Pop your USB stick in your computer if you haven't already and open the Rufus tool you just downloaded. Now follow these steps (click on the images to make them larger, if required):

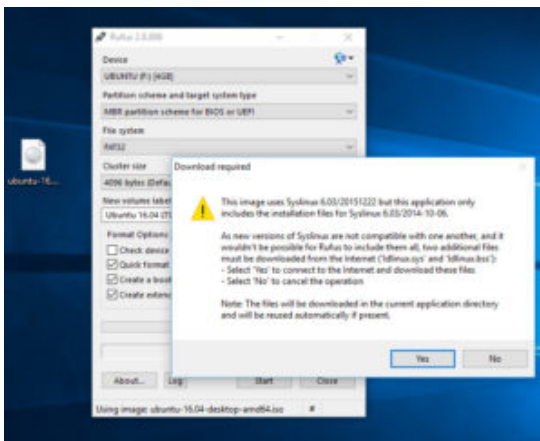
1. Choose your USB stick from the dropdown entitled 'Device.'



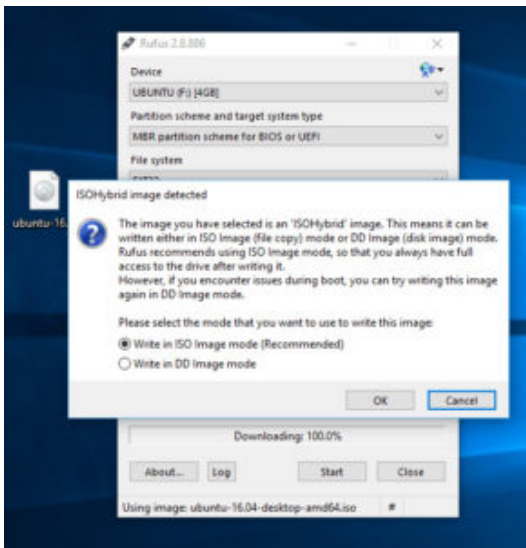
2. Further down on the right hand side you'll see a CD icon with a little box under it, click on that and an 'Open' dialog box will appear, locate the ISO file that you downloaded and then click **Open**.



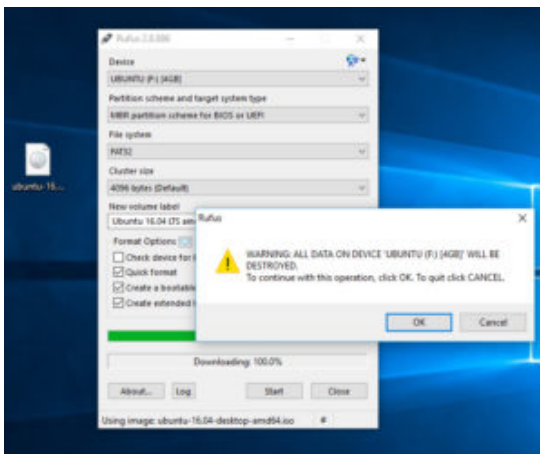
3. Click on '**Start**'
4. You'll be asked if it's OK to download the SYSLINUX software. Click '**Yes**'.



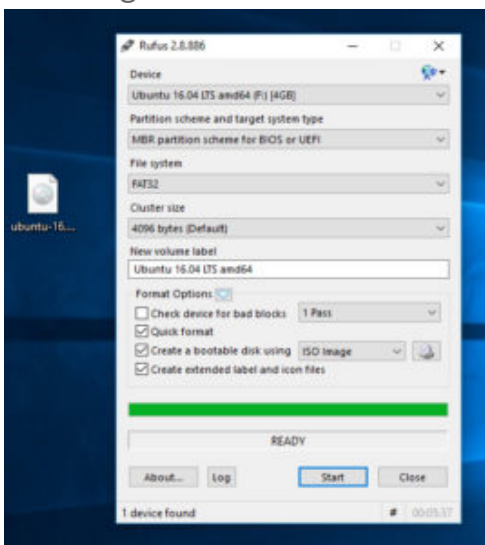
5. Once that's done, you'll see a dialogue about an ISO Hybrid image being detected. Simply click **OK** (*Write in ISO Image mode*).



6. Confirm that you selected your USB stick and not another one by accident! Take note that any data you have on the USB stick at this point will also be deleted. Press **OK**.



7. Once it's finished, restart your computer, as detailed a little more in the following section.



'Burning' the ISO file to DVD

If you prefer to use a DVD rather than a USB stick then you are in luck, burning an ISO disk image to a DVD is easy these days if you have Windows 7 or later, then simply right click on the icon of the file you downloaded, which will be named something like **ubuntu-14-04-desktop-amd64.iso**. Once you right click the icon, you will see the option 'Burn disk image'. Select that option and pop a blank DVD-RW or into your PC and click burn. If you would like further instructions on this process or are using an older Windows or other operating system, [check out this easy guide at the Ubuntu web site](#).

Ready to install!

Hopefully you are now armed with a Linux DVD or USB stick that's good to go. If the DVD was ejected from the CD player, pop it back in the drive and **Restart** your computer. Most PCs will automatically try to start the computer from the CD drive or even USB, so hopefully you will be presented with a welcome screen after a minute or so. If you don't see this, or if your PC started up in Windows instead, make sure your PC is set to boot from CD (or USB) before any other disks. You can change this setting in something called the BIOS or Boot Order setup. Often when you start a PC you will see a message like 'Press F10 to Enter Setup'. Hit that key and enter the setup, you should be able to change the boot order to something like CD/DVD or External USB Media, Save the settings from there and then restart once again.

5.4 The Install Process

Okay, it's time to put the DVD or USB stick in the computer and restart it. If you don't already have an Ubuntu CD, visit Ubuntu's Download site and download the ISO image of Ubuntu. It's around 800MB in size, so you will need a 800MB CD-R or DVD-R and a suitable DVD-RW drive.

If you don't have a DVD/CD-R Burner or if you have a slow connection then you can [order a copy of Ubuntu from Canonical](#).

I've devised the following video tutorial to walk you through the process of installing Ubuntu Linux 13.04 on your home PC. It assumes that you already have Microsoft Windows pre-installed, so it shows you how to re-partition your hard disk to put Linux on it. Note that although the version is 14.04.01 LTS, the steps are practically identical for any modern version of Ubuntu or Linux Mint.

Part 1 of 2 (Guides you how to download and install the Ubuntu 14.04.1 LTS iso image and make it bootable on a USB stick):

<http://youtu.be/Zz7rpuAmbCo>

Part 2 of 2 (now you have set your PC up to boot from USB and set up the USB stick, it's time to boot the Ubuntu Linux installer from that USB stick!):

<http://youtu.be/jWnMTvcoK5o>

5.5 Notes for Installation

Username and the Administrative (root) user

When you are installing Ubuntu you need to set up a user account (as demonstrated in the above video tutorial). Something to note about this first user on your machine: Ubuntu always sets the first user specified as an administrative user. In this context, this means that you, the first account on the machine (you can have as many as you like), will also get administrative privileges on the machine to do things like install software and deal directly with hardware and all things associated with it. Take this information with a little caution. If you are asked again for your password when doing something in Ubuntu, it is asking you to escalate your own privileges into what Linux calls the 'root' user. Root is simply the username of the administrator in Linux, as administrator you have free reign over the system at all times. Do not perform tasks as root unless you know what you are about to do, or unless you have good confidence in the task ahead!

A little word on passwords

During the install process you'll be asked to select a password. It's important that you select a strong password, because in time to come, you may wish to open up services such as remote access onto your machine. It's a simple step but believe it or not, still a reasonably effective method of security, do not choose a simple word as a password. Choose random things, like for example, your favourite colour, and your first car, with a few numbers (maybe your year of birth) sprinkled in the middle for good measure. Here is a pretty strong password and could be pretty memorable to the right person:

blue77volvo240GLS

No, I wasn't born in 1977, my favourite colour isn't blue, and I've never had a Volvo 240GLS. However, you get the idea. The password is still important, especially if you ever run any server or sharing software on your machine in the future.

5.6 Finishing Up

By now, most of the software you will need will have been copied to your new partition on your hard drive, your user account will be set up and your regional settings are all ready for you. It's time to restart the machine. Make sure the CD is removed from the drive when prompted and continue onwards!

Welcome home!

When you start up your computer, you will likely see a boot-up screen asking you if you wish to choose Windows or Ubuntu (that is, if you installed Ubuntu alongside Windows). Choose Ubuntu from the list using the cursor keys (the up/down arrow keys) and hit enter and it will start up Ubuntu.

The next screen you will see is the Ubuntu login screen. It's the screen you will see every time you start up Ubuntu. If prompted, select the username that you created earlier (or type it in if required), then enter your password. The system will then log you on.

5.7 The Ubuntu Desktop

The Ubuntu desktop is a friendly place, which we will cover in later chapters. If you are used to Windows, your 'Start' menu is the bar along the left hand side. The top button which has the Ubuntu logo on it allows you to search your computer for content (Apps, documents, audio etc) simply by typing it's name, but if you don't know what App you are looking for, Click on the Ubuntu icon, note the little 'A' icon down at the bottom of the screen, click on that and it will show you 'Recently Used', 'Installed' and 'More Suggestions' for Applications. Beside the 'Installed' line you can see it reads 'See 76 more results'. Click on this and you can see all of the apps installed on your Ubuntu desktop. This same action can be done for documents, music, photos and movies with each of the respective icons.

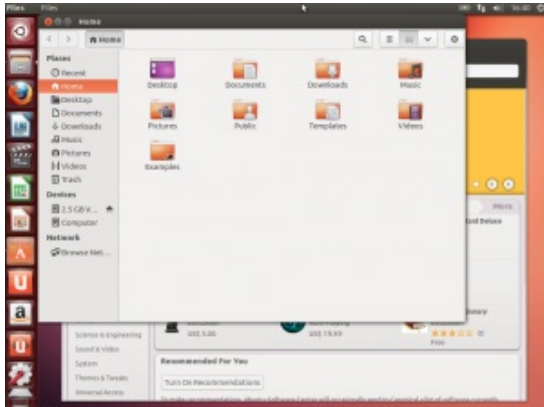
Favourite Apps

You'll note that a few favourite Apps have already been 'pinned' to the bar on the left, these include the Firefox web browser, the LibreOffice (free Microsoft Office compatible word, excel, powerpoint suite). You can pin your own favourites to the left-hand side bar if you desire simply by dragging an icon from the above Applications search list into the bar.

Where are all my files?

On the left-hand side bar there's a filing cabinet icon. Clicking on this will show you all the files on your computer. If your Ubuntu machine has been set up to see your

Windows file systems, you will also be able to see your the files you work with in Windows from here, just look under 'Devices' on the left of the file browser.



A screenshot of the Ubuntu 14.04 desktop with the file browser open. Note the 2.5GB Volume on the left hand side. This is similar to what you would see if you have a Windows file system available to you.

Chapter Six

How do I get software for Linux?

Okay, you've installed Linux, you've chosen your GUI, and you've picked up a few basic commands. You're probably feasting for some software now. Linux has an abundance of software out there, mainly available over the net, mainly for a zero fee. This chapter covers how and where to get software.

Check out Chapter 6 on the website

How do I get software for Linux?

6. Software for everyone!



open source

As discussed in later chapters, software in Linux is often free of charge, some of the most famous software titles around are available for Linux. Most Linux distributions these days offer simple ways to download and install software which are bundled up and ready to use in 'packages'. Sometimes, other software can be obtained in other formats too, so this chapter talks about all of the different ways you can use and obtain Linux software.

This chapter talks a little about the theory behind how software is distributed, and why. If you intend on using Linux for a while, it's probably quite beneficial to know this, however if you are just getting started out and want to install some software, jump to [Chapter 7](#), to find out how to install software .

6.1 How software is distributed in Linux

If you are a Windows or Mac user, you will be used to software being delivered to you in binary format. In Windows, the most common binary format you will come across is the **exe** file. A binary file is a file that has already been compiled from the original source code, to a binary (or closed source) format. You can't see the code inside the binary file, to the human eye, it's a big blob of random gobbledy-gook, that only your computer can read. This is done by the application developers (or programmers) of the software. It makes it ready for your operating system, and it also usually bundles in a lot of ancillary software which help it work. This ancillary software are commonly referred to as 'libraries'. Some times this can make the software more 'bloated' than it might otherwise need to be, and it doesn't let other

people see bugs or security issues in the software.

Linux, on the other hand began with a guiding principal that software should be free, as in free speech (not necessarily as in cost). To make software open (read free/libre), software authors needed to ensure that the software that was shipped was also made available in the original source code format, so that if you felt like contributing to the software, whether to fix a bug, add a feature, or make a derivative (fork) of the application, you could do so at will. As long as you too, kept the source code freely available. This is the guiding principal behind the [GNU GPL \(General Public License\)](#), and is the license that is most commonly used with Linux based software.

For ease of installation, most of the software you will obtain in Linux these days though, will come as 'packages'. These are pre-compiled bundles of software. The source code, however, is always available as a companion package, should you wish to see it.

Tux Tip: Note that if you wanted to turn the source code (written in human readable text form), into usable software for your computer (a binary), you have to use an app called a [compiler](#). This is an advanced topic which will not be covered in this chapter.

6.2 A little word on Dependencies

Feel free to skip this section if you aren't interested in the details about *why* Linux software packages are a better idea than big binary files found in Windows and MacOS.

As noted earlier, in Windows and Mac OS, a lot of software comes in one big binary file (a 'blob' if you will). Most of the other tools or software that it needs (called dependencies), are usually bundled up inside those binaries. Sadly though, sometimes the same libraries are already installed on the operating system, put there by other applications before. This means that you might have the same or different versions of libraries kicking around on your PC taking up space and making it slower than you'd probably like.

Linux packages are often a lot smaller, because they rely on the other dependencies from other software to be installed only once. Take for example, you might want to install a hypothetical app called 'liarfox'. It's available for Windows and Linux. The liarfox package might be 75MB in size in Linux, versus a 250MB in

Windows. When you install liarfox in Linux, it also says it's going to download another 100MB in other packages too, making the total installed on Linux 175MB. Still a little lighter than Windows. But the good thing is that in the end, it didn't need to download all 100MB of extra packages. Why? Because they were already installed and in use by other software on the system. The *dependencies* had already been met.

6.3 Using binary packages

Software in Linux today is packaged up for easy installation in most cases. The most common formats are as follows:



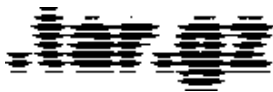
.rpm files: Redhat Package Manager

Software that is in Red Hat Package Manager format is used by Red Hat, Fedora Core, SuSE and CentOS variants of Linux.



.deb files:

Debian package formats are used by Debian GNU/Linux, Ubuntu, Linux Mint and other variants.



Tar packages can contain pre-compiled binaries for specific systems, or can also contain source code. If you have used ZIP files or RAR files in Windows or MacOS, then tar.gz files are essentially the same thing. Anything can be stored within them.

In the case of deb files and rpm files, most modern day Linux distributions have what is known as a package manager. This means that you don't need to worry about downloading or even dealing with installing individual packages, and their dependencies. The packages (.deb and .rpm files) are installed for you, at the click of a button (or the type of one command).

6.5 Obtaining the software that came with your distribution

Back as far as 2005, Ubuntu Linux introduced the Ubuntu Application Store. Note that this was before the iPhone and definitely before the Apple 'App Store'. It's a seriously easy way to search for, download and install software. App stores are the most user friendly way to obtain software in Linux these days. Most of the popular Linux distributions have an equivalent app store. For example, Fedora uses a [menu option](#) that conveniently says "Add/Remove Software".



Ubuntu Software Centre shows how easy it is to download and install software in Linux.

This almost sounds too good to be true, but there is no such thing as a perfect world! Exceptions exist. For the software title that you require to be available via the 'App Store', means that two things have happened:

- a) The Linux distribution vendor (eg, Canonical and/or Debian, in the case of Ubuntu), need to be aware that the software exists.
- b) it's been vetted officially to be represented in the 'main/approved' distribution channel (repository) of the Linux distribution. Note, If the software is acknowledged by the distribution vendor, it may still be available in an alternate distribution channel, which may have to be additionally enabled.

If both of the above cases are not satisfied, then, like in the Windows or Mac OS world, you need to go off and google for the software vendor's website and find the download yourself. We talked earlier in this chapter about dependencies. When you download 'unofficial' software like this, and you install it, it can't be guaranteed that it will work with your Linux distribution because the software it depends upon (libraries, often) may not exist within your Linux distribution. If you ever hear of the term 'Dependency Hell', this is what people are talking about!

Tux tip: If you are installing software from websites, you'll often download it as .deb, .rpm or even .tar.gz format. You'll need to double-click the downloaded file to launch the installer, or with .tar.gz, decompress the file and go to a command line to install it from there. If you want or need to install deb or rpm packages from the command line, you can also do that too, more information on that in [Chapter 7](#).

This matter of obtaining 'official' software from a repository can often be difficult to grasp for first time Linux users, as it's a different way to get software, however thanks to commonality these days of the Mac OS App Store and the Microsoft Store, it's getting to be a more understood way to obtain software.

6.6 Okay, I got the software, Now how do I install it?!

A- ha! For that, you'll need to move on to the [next chapter](#)!

6.7 Advanced Topic: Other ways of getting software

This is an advanced section for those of you that want to get software from other sources/websites. It is not required reading for most Linux users, or new users but is here for completeness sake.

.tar.gz

As mentioned, there is the traditional .tar.gz format. Most of the time, you will find that the .tar or .tar.gz format is used to zip up source code format software, a .tar.gz file is often referred to as a tarball. Tarballs are double-archived files. This means, that they're first archived with a tool called tar (the standard unix archiving tool), and then to zip or compress the file, the tar file is then zipped up with gzip (the GNU Zip tool, which is akin to PKZIP or RAR).



Tux Tip

You can still unzip/zip WinZip/PKZIP compressed files with Linux, using the unzip and zip tools, rather than gunzip and gzip.

Furthermore, the popular desktop systems, including KDE and GNOME all have a GUI tool built in to handle .gz and .zip files effortlessly!

To decompress (unzip) a tarball at the command prompt, type: `$ tar zxvf filename.tar.gz`

If you are using Arch Linux, the officially bundled binary packages are distributed in .tar.gz format, although they are usually installed using the pacman package manager.

Slackware Linux also bundles its pre-compiled binaries in .tar.gz files, but in a specific archive layout.

ebuild

The ebuild format is a specialised bash script which is used by Gentoo Linux which automates compilation and installation of software packages. It is used within the Portage Software Management system, as well as the emerge tool.

SNAP

Snappy is a new package management system made by Canonical for Ubuntu phone operating system, as well as Ubuntu Desktop and package most dependencies within one package. It is now available in other distributions including Arch and Fedora, and could become the standard package format in the future.

Other non Linux packages: PKG, APK, APPX

For completeness, other operating systems have made packages, or are starting to use packages:

The PKG format is used by Apple on the MacOS, iOS platforms, generally transparently. It is also used by Sony on the Playstation 3, and in other Unix like operating systems, such as Solaris, UNIX System V, Symbian and BeOS.

The APK format is the Android package format and is used on Android mobile phones, tablets and other smart devices.

The APPX format is used by Microsoft Windows 8 and above, as well as Windows phone, mainly within the Microsoft Store, transparently to the user.

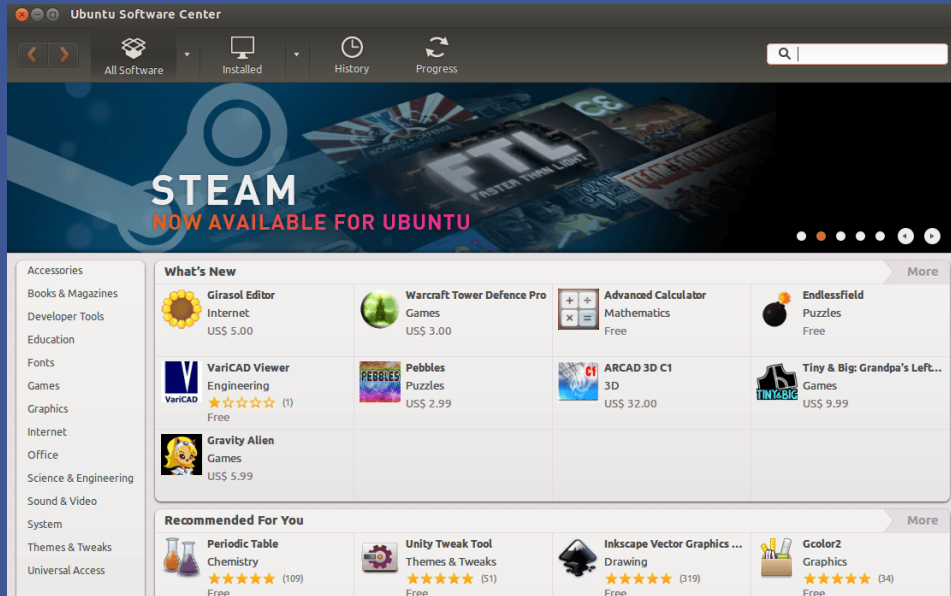
Still can't find the software you are looking for? Use

Google!

It's happened to the best of us, and in time, it may happen to you. If you are looking for a package, but you can't find one on the authors web page, nor on sites like rpm.pbone.net. You can only find the tarball and you just can't find it anywhere. Well, you would be surprised at who actually makes up these packages from tar files. Mainly you'll find that distribution vendors package them, but you'll find other people out there doing it for the hell of it, to help you and the rest of us out. This generally happens if the program is a dog to compile as a tarball.

A word to the wise though: If you are downloading an RPM or DEB package from a third-party, it could have been tampered with, or worse, wreak havoc with your Linux installation - use third party binaries as a last resort and with caution!

Chapter Seven



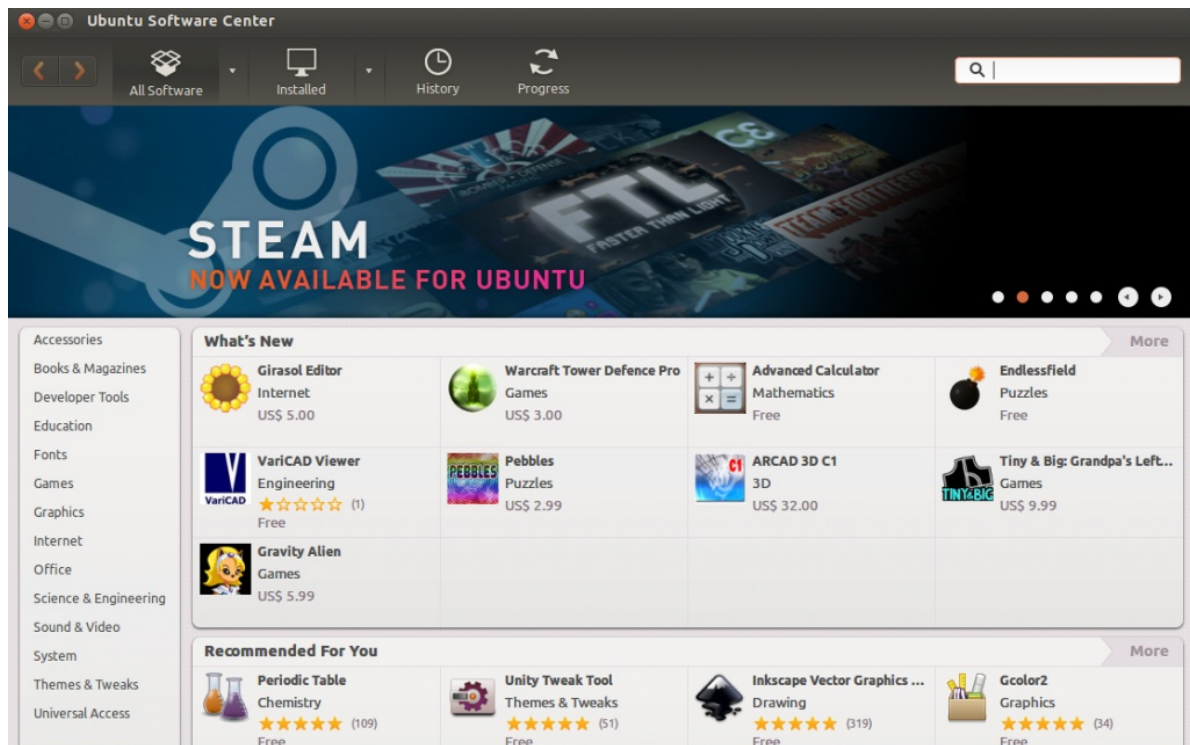
How do I install software?

You've downloaded the software that you want, and you discovered that you don't know how to install it. This chapter covers installing .RPM, .DEB packages and tar.gz files.

Check out Chapter 7 on the website

How do I install software?

7.1 The easy way: Ubuntu Software Center etc



Ubuntu Software Centre shows how easy it is to download and install software in Linux.

There are many different distributions (flavours) of Linux, as demonstrated in [Chapter 3: Choosing a distribution](#). Each distribution of Linux handles the installation of software slightly differently, however as we saw in [Chapter 6](#), they all use one of three main 'packaging' types, so they rarely vary in drastic ways.

When Ubuntu Linux first came out back in 2005, it used the 'Synaptic package manager' as an easy-to-use tool to obtain software. You simply searched for the software you wanted, clicked 'Select to install' beside the package you wanted, then pressed Apply. It wasn't hard to do, but since then, Apple made the App Store with the advent of the iPhone etc. Since then, the Mac, the PC (Windows) and Android mobile also got an App Store, so it's only fair that some polish went into Linux too. The old synaptic package manager can still be used if so desired,

however, if you use Ubuntu, you will likely prefer the Ubuntu Software Center. It shows you which software you have installed on your Ubuntu PC, as well as all of the software titles available from Ubuntu. It even recommends software that you might like, based on the software you have previously downloaded.

Installing the software in the Ubuntu Software Center is as easy as clicking on the title you are interested in and tapping 'Install'. The rest is done for you, and the application is available from the Ubuntu Launcher (that brown circle icon up at the top left-hand side of the screen).

The rest of this chapter talks about how to install software utilising traditional software packaging methods, including apt-get, synaptic package manager and YUM.

7.2 Installing .deb (Debian packages) and the APT Tool



The Debian package management system is a very well made software packaging model. It has similarities with the Red Hat system (RPM), however, when DEB packages are combined with the APT tool, both the act of obtaining software, and issues with dependent software are almost completely removed.

The DEB/APT system is highly popular, and is found on many systems other than Debian, including the Ubuntu, Linux Mint and more.

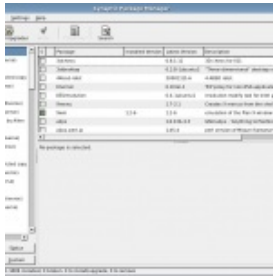
These are the main parts to the DEB system:

- dpkg - A command line program with which you can install .DEB packages. This is the most basic (and most difficult way to install debian .DEB packages)
- apt-get - An easy-to-use command line tool that offers a simple way to install packages, and unlike dpkg, does not work with the .deb package, but uses a file found in /etc/apt/sources.list to obtain the relevant .deb file from the net.
- dselect - A text-based menu driven interface that acts as more than just a frontend to dpkg. Allows for installation and removal of packages
- Aptitude - An [ncurses](#) terminal based front end to APT. It's popular for it's user friendly interface and highly descriptive nature.
- Synaptic or Adept - The graphical frontend tools for GNOME and KDE respectively that provide an easy to use interface to apt-get. They make

installing software easier as you don't have to remember any commands, which most new users will feel more comfortable with.

Now that you know what the main Debian DEB tools are, we'll step through installing a Debian package first using the Synaptic package manager, which is the default tool for Debian Linux, and then we will briefly cover installation at the command-line shell:

Installing software with the Synaptic Package Manager:



Synaptic Package Manager. Click thumbnail to expand

To start up Synaptic, click on the System menu at the top left of the desktop menu.

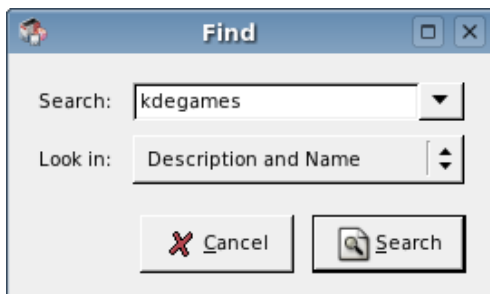
Then Click on Administration.

Finally, Click 'Synaptic Package Manager'

Once you type your administrator password (which is the same password as the main user of the machine), you will shortly be presented with a screen similar to the one on the left.

Amongst the many options you see, there is a large list on the left hand side of the window.

This list contains all sorts of different categories of software. If you don't know what you are looking for, start here!

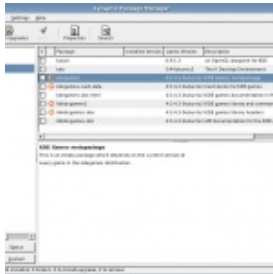


Searching for packages in synaptic

By clicking on the Search button at the right hand side of the toolbar, it is possible to type in the name of a program you know the name of.

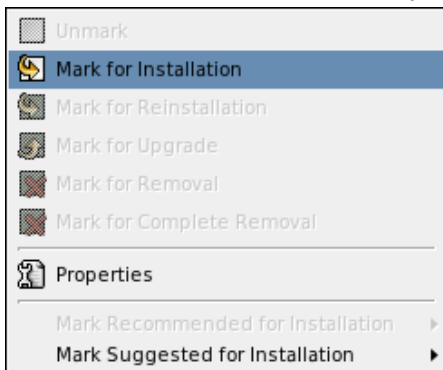
Note I am typing 'kdegames' here, because I know that I want the software package entitled kdegames.

By pressing the search button in the last step, I have now been shown the package 'kdegames' as well as a description, and version information.



If you are sure you want to install this title, then right click on it. The menu on the right will appear. By choosing 'Mark for Installation', the software 'kdegames' will be chosen for installation, when you apply these changes.

The example of 'kdegames' was specifically chosen because it is an example of software that has dependencies on other software (it needs other software titles to be installed for it to work). Synaptic/APT is showing you this



fact and is conveniently telling you that it will also download the extra software shown in the menu to the left to 'satisfy' these dependencies.

You should click 'Mark', in order to continue.

One you click 'Mark', Synaptic will take you back to the main screen so that you may choose more software to install, if you wish.

If you are ready to install the software you chose earlier, click the 'Apply' button.

That example would install the most recent version of the program 'firefox'. Apt would also tell you that it needs to download some other software (dependencies) in order for 'firefox' to run. A few other commands for Apt you'll need are:

```
apt-get update
```

Updates the APT source information, to tell it about any new software in the APT repositories.

```
apt-get upgrade
```

Upgrades any old software on your machine to the latest versions automatically.

```
apt-get dist-upgrade
```

Upgrades the distribution to the latest available version of the distribution.

```
apt-get remove
```

Removes from your system, and any non-required dependencies.

Installing .DEB packages at the command-line shell:

Firstly, download the .deb package and pop it into any folder on your system, then simply install it by running the following command using the terminal:

```
$sudo dpkg --install package-name.deb
```

That should be the package installed, although again, there are dependencies to think about, and as with RPMs, make a note of any dependency errors, download the appropriate DEB package(s) to meet the dependenc(ies) and try again.

7.3 Installing RPM files (red hat packages)



Red Hat, one of Linux's first distributors came up with a neat solution to the problems .tar.gz files and compiling has for the normal user. They pre-package the tar file, zip it up and make it do all the hard installation work for you. This system is called RPM and it's the standard software installation method for a lot of Linux distributions today, such as Red Hat, Fedora, SuSE and Mandriva.

If you are unsure if you have an RPM system or you just want to check what version of RPM you are using, then try typing the following at the [Linux terminal](#):

```
$ rpm -q rpm
```

You should get a similar answer to this:

```
rpm-4.0.3-5
```

If you get something like 'command not found', then it sounds like you don't have RPM installed, you may be using a Linux distribution that does not use RPM natively, for example, Debian, Ubuntu, Gentoo, Slackware, Mepis or Xandros have a look at the DEB or TGZ sections of this page.

It's important that if you go to download an RPM from the net, always try to get one that was packaged by the vendor of your distribution.

For example, if you go to the web and search for an RPM package called 'firefox', and you get 3 RPMs back: One from Red Hat, one from Mandriva and one from SuSE. If you have a Mandriva Linux distribution on your PC, make sure you use the Mandriva one.

The reason for this is all down to fitting into your system configuration structure and things called libraries, which vary from distro to distro.

How to install the package

Okay, let's presume that you have an RPM file ready to install called netscape-4.76-3.i386.rpm

You can install it in the following ways:

At the terminal/console:

```
sudo rpm -Uvh netscape-4.76-3.i386.rpm
```

Preparing 100%

Installing 100%

The options -Uvh stand for the following:

U - Upgrade package if already installed, or install if not installed

v - Be verbose about the installation

h - show hash symbols to indicate progress of installation

Don't use RPM - YUM is easier and better:

YUM (mentioned in the previous chapter) is a system much like Debian's APT, but for Fedora and other RPM based distributions. It makes dependency problems far less likely for Fedora users.

Installing an RPM package through YUM can be done by the following steps using the [Terminal](#) application:

```
$sudo yum install netscape
```

Note that you do not need to specify the version of software you are installing. YUM goes out to the Internet and automatically pulls down the latest version it can find of 'netscape', and installs it for you, along with any other software you may need, in order to run 'netscape'.

What if I don't want to type commands in to install software via YUM?

Then use the graphical program, Yum Extender (or similar)!

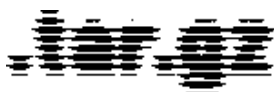
How do you install it, I hear you ask?

```
$sudo yum install yumex
```

Now you can access Yum Extender in the "red hat" menu, under "System Tools."

For further information on YUM, see [This page](#)

7.4 Installing .tar.gz. files (tarballs)



.Tar.gz files, or, Tarballs stand for tape archive and are the Unix equivalent of zip files for the Windows world. They can contain any files, but are often used to package up source code for programs.

Tarballs come packaged in five main flavours:

.tar (standard tar archive with no compression.)

.tar.gz, .tgz (standard gzip compressed tar archive. same as .tar.gz.)

.tar.bz2 (tar archive compressed with extra compression tool, bzip2)

.bin.tar, .bin.tar.gz or .bin.tgz (less common tar archive that contains binary files rather than source).

Most of the time, you will be dealing with .tar.gz files.

Here's how to extract (unzip) a .tar.gz file in two different ways:

At the [Console](#):

```
$ cd /directory_that_the_tar_file_is_in<br />  
$ tar zxvf tarfile.tar.gz
```

To explain the latter command, tar decompresses the gzipped file (with the z flag, which is short for gz, or gzip), x means to extract, v is for verbose (so you can see what's happening as it extracts) and f means extract the following file (in this case tarfile.tar.gz). Remember that tar was originally used for extracting archives from tapes, back in the old days, so by default it expects the standard input to be a streaming tape archive.

To extract a tar archive In Gnome or KDE (Graphical Desktop):

Right click on the icon for the appropriate tar file in your file manager.

Choose Extract (or in KDE, choose Open with Archiver).

Extract with the tar file with the relevant archiver program.

Okay, now you've extracted it you have to either:

- a) Compile the source code you just extracted
- b) Run the installer script which is part of the files you extracted

So, how do you distinguish whether you have just extracted a tar file with source code in it, or whether it's a binary, with an installer in it?

Usually, the contents of the .tar.gz file will help you out here - A file containing source code, will often contain a file called 'Makefile' somewhere in the first folder within the extracted volume. This file is used to compile, or, make, the software.

A tar file which does not contain source code mainly holds a binary installer file in it, the filename of the installer usually ends in .sh or .pl.

For example, the program VMWare, contains a program called vmware-installer.pl in the extracted root folder.

To run the file, you usually need to give yourself 'permission' to run it:

```
sudo chmod 755 vmware-installer.pl (changes permissions on the file so that it can be read, write and run- executed)
```

```
sudo ./install-vmware.pl (runs the installer)
```

If you found a 'Makefile', then you need to compile the source code. Here's how to do it:

Most of the time, you will need to use the terminal to compile source, so use an [xterm/console/terminal](#) and go into the directory that has been made by the package, eg:

```
$ cd /directory_that_the_tar_file_is_in
```

```
$ ls -l
```

```
Total 302
```

```
-rwxr--r-- 1 user group 2907 May 21 17:15 mytarfile.tar
```

```
-rwxr--r-- 1 user group 0015 May 21 17:15 newdir/
```

```
$ cd newdir/
```

at this point, make sure you read the INSTALL file. You'll find that almost every tarball that you download (especially GNU software) has at least a file called INSTALL, COPYING, README and CHANGES

Most of the time the INSTALL file says the same thing, it's a generic process for installing tarballs, but if a program requires to be compiled in a special way, you'll find out in either INSTALL or README. If it's helpful, it will tell you the names and websites of any other software you will require to download in order to install this software. These other pieces of software required are called 'dependencies'.

If you were installing a generic program, extracted from a tarball and presuming that we just changed into our directory, as above, we could do the following to compile the program:

```
$ ./configure
```

(Take good notes here for any configure errors)

```
$ make
```

(Take good notes here for any compile errors)

```
$ make install
```

(Take good notes here for any compile errors)

```
$ make clean
```

(this cleans up after a successful compile)

Why compiling is a pain in the ass (for most people), and the problems you might have.

The above procedure doesn't sound too difficult, and in theory, it shouldn't be. But it doesn't always work.

Most of the time this is because of dependencies on other programs, you need other software (usually programming libraries) to be installed first, in order to compile this software.

Picture this scenario: You attempt to install tar.gzipped game called xtux.

The `./configure` bombs, and you noticed on the web site of xtux, and also from the output of the `./configure` something about SDL. You're not quite sure what it is, but you go onto a site like google anyways and type in SDL.

You find out that SDL is in fact a popular graphics library for X and that it's necessary for xtux.

You download `SDL-1.2.3.tar.gz` from the SDL website and install that tarball. It installs fine, so you try installing xtux once more. It still bombs out, but this time it gives you a different message: `could not find Qt equal or greater than 1.3 on a ./configure`.

You check your system for QT version 1.3 or greater. You have 1.3.4 so you should be fine. Why is this error coming up? Well, it's probably because Qt (which is another graphics / programming toolkit) is installed, but is not in the folder that ./configure is looking in. You can edit ./configure yourself to see if there is anything you can do to amend the situation yourself, or try removing Qt, and installing another instance of it from another source.

I find that RPM based distros often put stuff like Qt in places that a normal tarball doesn't, so that's often the reason for these compile problems, make sure that if you have installed the normal version of an RPM (binary version), that you also install it's accompanying -dev RPM if you wish to compile .tgz based source against it. By this I mean:

Make sure you have installed qt.i386.rpm and qt-dev.i386.rpm if you are compiling something that relies on QT, as the -dev package will provide the qt source code to the source program you are installing.

Dealing with your files and programs

If you've installed your tarball, RPM or DEB package and you want to run it, or perhaps you are interested in knowing more about the Linux file system including dealing with permissions, [have a look at the tutorial on 'managing my files'](#).

Alternatively, If you want to know more about using your desktop every day, for office use, for multimedia and more, continue on to [Chapter 8](#).

Chapter Eight



Using Linux every day for work and play

Linux offers so many different uses, and finding out how to do it all can be a difficult task. This chapter makes it easy to find out how to do all the things you want to do with your computer in one easy page. From listening to music, watching TV, using office software and even playing games, this chapter covers all the standard desktop uses of a home or small office PC.

[Check out Chapter 8 on the website](#)



Using Linux Every Day

What would be the use in Linux if it didn't have some great software? I'd rather put up with a poorer Operating System than have no software, wouldn't you?

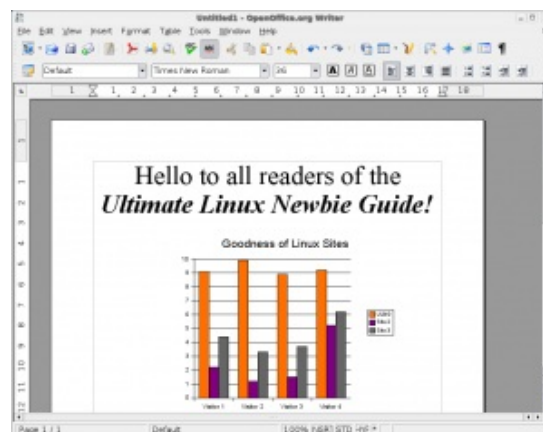
Fortunately, Linux is teeming with great software for all sorts of purposes, and most of it is free of charge. This chapter discusses the most popular uses of a modern-day PC, and where Linux fits in with all of this.

Using Linux for all your office needs

When Linux first came out back in the early 1990s one of the downfalls of Linux was that good office software was lacking, making it an impractical system for business users. From the late nineties onwards, this is no longer the case. Software like OpenOffice and KOffice are excellent, complete packages, and standalone software like AbiWord and MrProject add further to the great selection of tools.

With respect to Groupware tools, either Novell Evolution or Kontact provide all the features of popular groupware offerings from Lotus, Microsoft or Novell.

Office Package Software



LibreOffice is an ideal replacement for Microsoft Office

LibreOffice: Microsoft Office users will be pleased to know that the most popular Office suite in Linux, LibreOffice, supports Microsoft Word Documents, Excel Spreadsheets and PowerPoint presentations.

The latest incarnation of LibreOffice includes a Word Processor (with export to PDF feature), Spreadsheet, Presentation creator and more.

LibreOffice was built on software made by Sun (now owned by Oracle) called OpenOffice and before it, StarOffice.

Although OpenOffice.org is the most popular office suite for Linux today, it should be noted that KDE sports KOffice, which is similar to OpenOffice in many ways, and features much of it's functionality. GNOME also has various office components such as AbiWord and Gnumeric. If you have an older computer with lower memory or storage, you may want to try one of these alternatives.

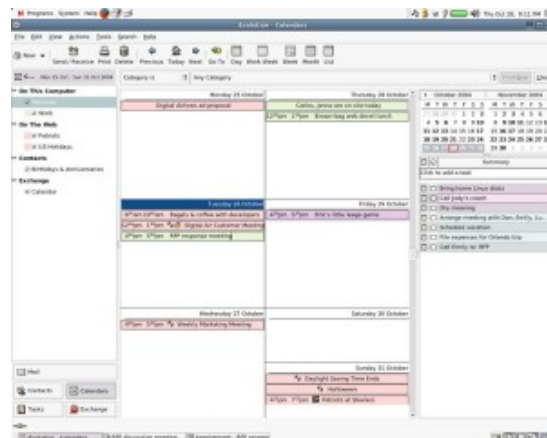
What if I don't want to use another office or other Windows app? - I like most of my Windows software just fine!

It is understandable that some users may not wish to move from what they know best. Even although OpenOffice is very similar in looks and features to Microsoft Office.

You may find solace in knowing that it is possible to run Microsoft Office as well as a great many other Windows apps on Linux! - Although it is not software written for Linux, it is possible to run it using software from a firm called CodeWeavers. The software, called [Crossover](#) supports Microsoft Office, as well as many other popular Windows titles such as Adobe Photoshop, Microsoft Visio, Lotus Notes, Apple iTunes and Dreamweaver.

Groupware Suites (e-Mail, Contacts, Calendars, Tasks)

Evolution



Evolution Groupware in use

Evolution is a groupware suite for performing all of your day to day E-Mail, Scheduling, Contact Management, Address Books and more.

Users can retrieve their E-Mail from a vast array of sources including IMAP, POP3, Microsoft Exchange and Novell GroupWise Servers.

It's overall look and feel is similar to that of Microsoft Outlook, but enhances on a number of features, especially where security is important. It also supports PDA devices and LDAP servers for great connectivity, wherever you are.

Amongst Evolution's competitors are:

Kontact



KDE's personal information manager management software is a fantastic groupware suite for Linux.

Kontact features a summary overview, a very capable E-Mail client, contact manager, to do list, journal, news reader, note taker, RSS/Syndication client and a mobile device

synchronisation tool.

Support for Novell GroupWare and Microsoft Exchange are not as mature as Evolution, but are pretty much there in terms of usability.

Mozilla Thunderbird



Thunderbird is part of the Firefox family and is as sharp as it's browser counterpart. It offers E-Mail with built in Junkmail filters, IMAP and POP mail server support, contact manager and address book. It is lightweight and easy to use, but not nearly as full featured as Kontact or Evolution.

Using Linux for Artwork

Artwork has often been the lair of Mac users. This is mainly for historic reasons these days - (Apple used to have an exclusive contract with Adobe). However,

times have changed, and Windows users enjoy many of the same fruits as the Mac users do. It's also fair to say that this software is often expensive, however some excellent software exists for Linux for vector and bitmap artists alike and most of it is free.

GIMP

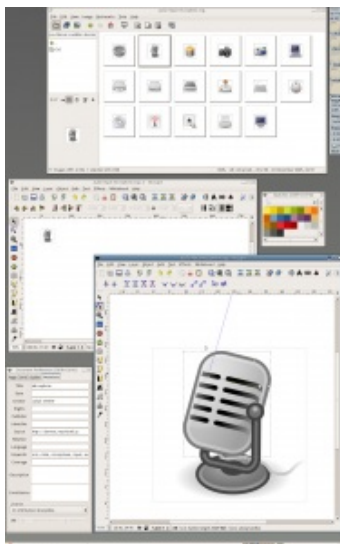


Gnu Image Manipulation Project (GIMP) is great for editing images

The GIMP is the classic Photoshop-like image editor for Linux. It's now ten years old, and can do most of the features of all the popular off-the-shelf competitors. In case you were wondering, GIMP stands for GNU Image Manipulation Project.

GIMP was made with tasks like photo retouching, image composition and authoring in mind, and it should be fairly easy for an accomplished Photoshop user to convert to the GIMP. If you can't live without Photoshop, it is possible to use Photoshop in Linux, using Crossover.

Inkscape



Inkscape Vector Image Editing

Inkscape is a vector graphics editor, similar to Adobe Illustrator, Freehand, CorelDraw, or Xara X using the Scalable Vector Graphics format.

Features include shapes, paths, text, markers, clones, alpha blending, transforms, gradients, patterns, and grouping. Inkscape also supports Creative Commons meta-data, node editing, layers, complex path operations, bitmap tracing, text-on-path, flowed text, direct XML editing, and more.

It imports formats such as JPEG, PNG, TIFF, and others and exports PNG as well as multiple vector-based formats.

Don't forget that OpenOffice.org also has a component called Draw, which has a reasonably good vector graphics editor. Also Kivio (part of KOffice), and Dia are

great alternatives to Microsoft Visio.

Playing and editing Music or Audio

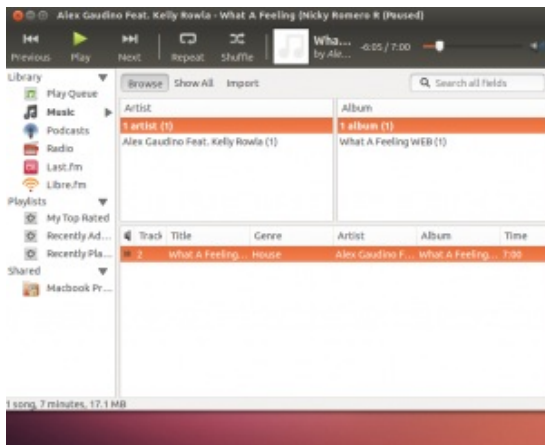
Linux has many applications for playing music and it's also really taken off as a platform for audiophiles. If you are a budding DJ, or play with MIDI instruments, Linux is a great operating system to use. There is a wealth of software available today which exceeds the requirements of many a Cubase user, and all for free!

Audio Players

The first audio player for Linux was XMMS. It is similar to the venerable Winamp product for Windows. XMMS suits as a straightforward no-nonsense audio player, which accepts many plugins and Winamp 2.0 skins, however it's getting a tad dated, especially when it comes to competing with the likes of iTunes and similar software.

As usual, both KDE and GNOME desktops have excellent offerings, and whilst there is a wealth of audio player software available for both platforms, I will limit the following reviews to just two players, one from each desktop platform: Rhythmbox (GNOME), and Amarok (KDE).

Rhythmbox



Rhythmbox

Rhythmbox is one of the many audio apps available for Linux, similar in interface to iTunes, it allows you to download/upload music to your ipod or other music device, as well as listen and organise all your music on your PC. It also connects you to Internet radio stations, Podcasts and allows you to transfer music to and from your iPod. It now has in-built support for playing, ripping and burning audio CDs.

amaroK



Amarok in action

amaroK, apart from having a stupid name is a great music player for KDE users. Fans of iTunes will be immediately relieved to hear that it looks very like iTunes and syncs with your iPod flawlessly.

It creates dynamic playlists too, like the party shuffle feature in iTunes, but better! amaroK features automatic CD cover finders for each album on your PC, so you know what the CD looks like when you play it. It also grabs the lyrics for every song you play on demand, as well as telling you pretty much all the info you would ever want to know about the band you are listening to, from Wikipedia. It also features a built in ID3 tag editor to sort out those rogue MP3s with invalid entries, and features MusicBrainz to take some of the guesswork out of it.

Banshee

Banshee is part of the GNOME project and used to be a default player for some Linux distributions. It's still very popular and works very like iTunes. It plays movies, syncs to your iPod or Android device as well as playing music.

Why can't I play MP3s, Xvids or DivX files etc?

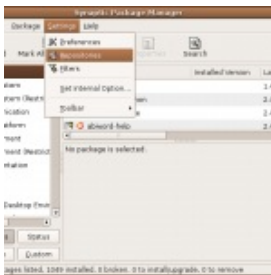
If you have just started using Ubuntu, you may notice that the default music player is called Rhythmbox. Furthermore, if you try to play an MP3 file, you will find that can't do it! This may be the same with the movie player when you try to play certain types of encoded movies.

There is a good reason for this - Ubuntu, as with Fedora, Suse Open Edition and any other free Linux distribution cannot ship with software that is either not free, or contains a license which is considered commercial. It may not be illegal to distribute such software, but you may be required to accept a separate license, or

use the software on different terms than the rest of the system. In order to keep the licensing system sane, this software is distributed separately from Ubuntu (as with other distributions).

You may have noticed when installing Ubuntu there was a tick-box which asked if you wanted to install "Proprietary codecs", if you ticked the box then, the software you require to play most movie and music formats should already be installed, otherwise, Here is how to add support for these 'codecs' after installation of Ubuntu:

If you are using the Ubuntu Software Centre, simply search for **ubuntu-restricted-extras** and click 'Install'. Otherwise, if you wish to use the Synaptic Package Manager, follow these steps:



- Firstly, open the Synaptic Package Manager.
- Next, add the Universe and Multiverse Repositories, by clicking on the Settings Menu and then clicking Repositories.



- Select the 'Multiverse' and 'Universe' Repositories so you can download commercially licensed materials. Click OK. Tick both of the bottom boxes, to add the Multiverse and Universe repositories.
- You will be asked if you wish to reload the package list from servers, Click Yes.
- Once the update has been completed, you will be returned to the main Synaptic screen, where you will be able to use the 'Search' button.
- Click Search and type **ubuntu-restricted-extras** in the text box. Right click on ubuntu-restricted-extras and select 'Mark for Installation'.
- You may be asked to 'Mark Additional Changes', which is fine. Click on Mark.
- Click on the Apply button (the green tick). You will be asked to confirm the changes, press the small apply button to continue and the packages will be

downloaded and installed for you.

If you need a refresher on installing software, refer to [Chapter 7](#) for detailed instructions on installing software for your particular distribution

See the [official Ubuntu documentation](#) for more info on restricted formats.

Sound Editing

Again, there are so many good tools out there for sound editing, it is hard to name but a few here, but we will try to keep it to a good few!

Ardour

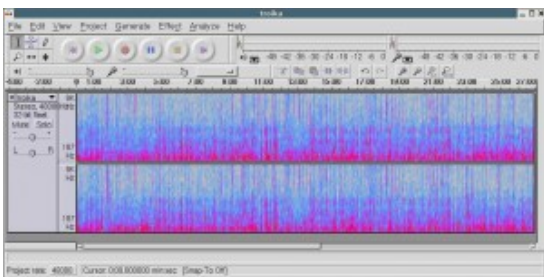


Ardour - a digital audio workstation for audio professionals

Ardour is a digital audio workstation which can be used to record, edit and mix multi-track audio. It even goes as far as mixing video soundtracks. Ardour is similar to ProTools in it's ultimate quest, and uses the much acclaimed JACK audio system.

Ardour has an excellent manual, and a quick and easy to use GUI. Ardour appears to be quick, snappy and doesn't use up too much CPU time, making it an excellent choice for a sound buff!

Audacity



Audacity is a fantastic, simple multi-track audio editor

Audacity is well worth a mention because whilst it does not have all of the features and power of Ardour, it does fill all of the needs of a sound-editing novice, whilst still beating any entry level sound editing software on the shelves.

Audacity's standard features can be picked up in a matter of minutes and allows for some really professional results. It offers unlimited tracks and many different effects in-built, so if all you need is straightforward multitrack or single-track editing, Audacity is a great tool.

Rosegarden

Rosegarden is a professional audio and MIDI sequencer, score editor, and general-purpose music composition and editing environment.

Rosegarden is an easy-to-learn, attractive application and ideal for composers, musicians, music students, small studio or home recording environments.

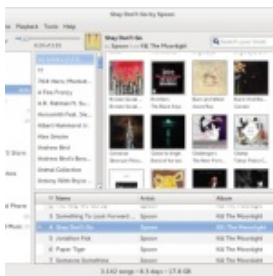
Sound on Sound magazine called Rosegarden the closest native equivalent to Cubase for Linux, which is no mean feat, and Rosegarden has come to the acclaim of many other publications such as Linux Format's Top Stuff awards.

[Click here for a list of some more audio file editors, DJ Software, Effects Processors and control applications for Linux](#)

Watching or Editing Video

most popular titles today are:

VLC - Multiplatform media player (Works on Linux, Windows and Mac).



Banshee Media Player for GNOME

Miro (It's also an online video content player, a Hulu or Netflix of Linux, if you will).

Banshee (an all-round media player for GNOME, similar to iTunes).

Kaffeine (KDE's answer to Banshee).

Some others to mention: NoAtun, SMPlayer, Totem.



VideoLAN/VLC is a great app, if a little difficult to navigate, however it offers a wonderful array of playback possibilities - handling almost every format out there, it's very versatile, but its real power comes from the fact that you can serve up media from any pc and get it to appear anywhere else - beam videos over the net, or just to another monitor in another room in your house. Support for resampling makes bandwidth problems with Internet broadcasting less of a problem.



Miro is a jack of all trades for Music, Movies, Online video, conversion and more

Miro allows you to convert any video formats you like as well as download and play almost any video. It will work with your current music library and will synchronise content to android and kindle formats. You can also buy music and apps inside Miro.

Watching DVDs and other Rights Managed Media Types

Just as with playing MP3s, DVDs and other formats such as DivX/Xvid, wmv and Quicktime all have their own proprietary plugins or codecs. See the above section about installing 'ubuntu-restricted-extras' if you need extra software to play these formats in Ubuntu.

Video Editing

Openshot

Supporting HD video, Blu-Ray and 3D, it's got a lot of features you might not expect from a simple to use video editor. If you are a fan of software like iMovie for the Mac, you'll feel at home with OpenShot.

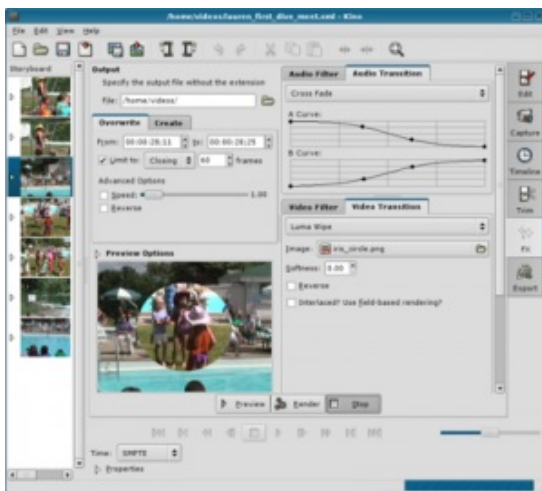
Cinerella



Cinelarra

If you need more than OpenShot or Kino, then Cinerella is for you. Hue, Saturation, Denoising, Compression, Time Stretching, Text-to-movie, batch rendering and much more are all staples of this sophisticated video editing suite.

Kino



Kino Video Editor

Kino is part of the KDE desktop apps, but will run with only the QT libraries. Kino features excellent integration with IEEE-1394 for capture, VTR control, and recording back to the camera. It captures video to disk in Raw DV and AVI format, in both type-1 DV and type-2 DV (separate audio stream) encodings.

You can load multiple video clips, cut and paste portions of video/audio, and save it to an edit decision list (SMIL XML format). Kino can export the composite movie in a number of formats: DV over IEEE 1394, Raw DV, DV AVI, still frames, WAV, MP3, Ogg Vorbis, MPEG-1, MPEG-2, and MPEG-4.

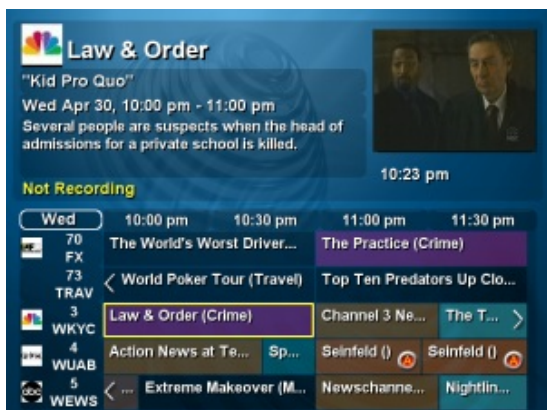
Blender

Although Blender is not technically a movie editor it's worthy of a quick mention. A number of movies have been created with this sophisticated 3D graphics and animation creator, it's been around a long time and is still in active development.

Watching and Recording TV in Linux

Recently, Linux has come to great acclaim in the TV space, due to the excellent range of software available at no cost. You can literally turn an old PC into a Linux style Sky Plus/Tivo box, all you need is a TV card and a big hard drive!

MythTV



MythTV allows you to turn your PC into a Media Centre complete with PVR and TV options.

MythTV is open source software that turns your PC into a PVR (Personal Video Recorder). It enables the user to pause live TV, Skip ads, use an electronic program guide, set recordings to record whole series of a particular program, edit recording schedules, organise and view your home photo and videocamera collections as well as listening to music and record content from the Internet, it excels at many points, making it a cut above the current offerings from Microsoft (Media Centre) and PowerCinema.

Also check out [MythBuntu](#), which is a pre-packaged distribution of MythTV, making it easy to install on a PC.

If you only want to do the basics, you can record and watch live TV then you don't need to use something as feature-laden as MythTV. Popular titles for viewing and recording TV are tvtime and xawtv.

The Hauppauge series of TV cards seems to work best with the Linux TV software as it is the most popular range, and best supported through the video4linux driver.

Playing games in Linux

Games have been a mixed bag with Linux, and still needs perfecting, however companies like id software and others are releasing their top titles for Linux as well as Windows these days, and is helping the overall popularity of Linux as a gaming platform no end.

Steam

Steam is a popular Windows (and laterally Macintosh) games platform. Made famous by games like Half-Life, Steam now hosts hundreds of blockbuster titles, and is now available for Linux. Not all of the titles are available for Linux yet, but this seems to be a work in progress.

Native Linux Games

Games such as the Team Fortress 2, Quake, Doom and Wolfenstein series are available 'natively' for Linux (that is, the software is written to work in Linux). Other games which will also work with Linux such as the Soldier of Fortune series have been ported from their Windows base. For more titles which work in Linux check out the [Games On Linux website](#).

The number of natively available Linux games is still low, however there is a solution, however - [Crossover](#) is a program which allows the user to play most of the Windows games titles within Linux. This software is not free, but is fairly priced, and offers a good degree of ease of use.

There are a great number of games which are also freely available Linux software which you can see at places such as [linuxgames.com](#) or [happypenguin.org](#). Tux Games have done a great job of porting commercial games to Linux which you can buy for great prices.

A video demonstrating how to get to grips with the GNOME Desktop

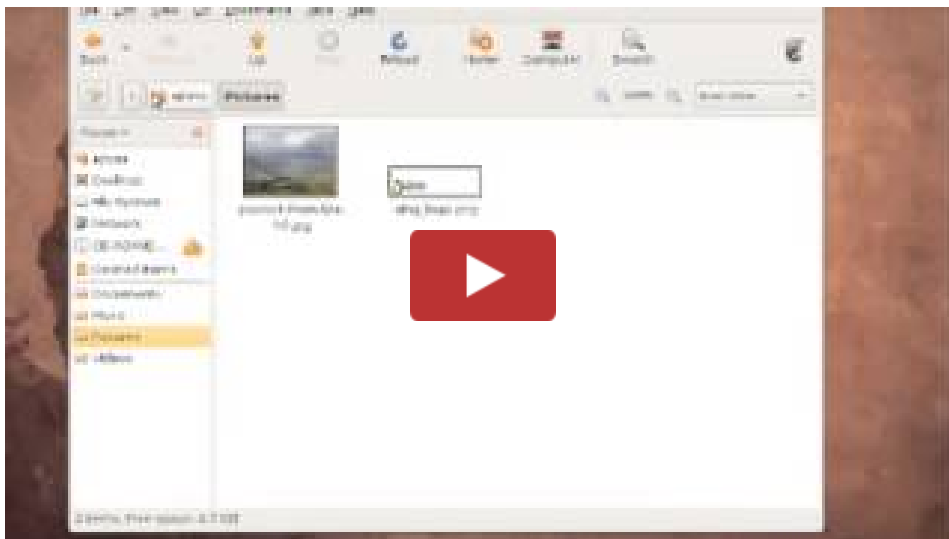
Now that you've read all about the various different applications you can use every day on your Linux desktop, why not see it in action! The following videos taken from our Articles & Further reading section show you each element of the Ubuntu Desktop so you can completely familiarise yourself with all that it has to offer.

Part 1 of 3: The basics of the basics of the GNOME Linux Desktop, Essential

applications such as text editors, photo editing, instant messaging, e-mail and web browsing with Firefox.



Part 2 of 3: Using the standard Office suite (OpenOffice/LibreOffice), how to navigate your files and folders on your computer and other computers as well as using the settings menu:



Part 3 of 3: Using Virtual Desktops and Installing Software as well as software repositories (sources) explained:

Quick Tips

The following pages have been selected from the website as an example of the useful selection of the tutorials on the website.

[Check out all of the tips on the website](#)



How to put a Linux ISO onto a USB stick and make it bootable on a Mac

So, you've got a mac computer and you want to put the ISO you just downloaded onto it. If you've used tools like [UNetBootin](#) before, then you'll have found it a nice easy graphical way on a PC. Unfortunately, Mac users have to go through a few more hurdles to get an ISO image onto a USB stick in order to make it boot properly.

What are we trying to do here?

So, just to be clear, you have downloaded Linux (say Ubuntu) from the Ubuntu website. When you downloaded it you get a .iso file. This .iso file is a disk image. In the past, we'd 'burn' this image onto a blank CD or DVD. They weren't really intended to be put onto USB sticks. USB sticks were more meant to pop regular files onto, not disk images.

However, this is 2016, and people don't have CDRW drives any more, so let's find out how to do this on your beloved mac. If you want a full tutorial on how to install Linux on a Mac computer, [check out our guide](#).

Step 1: Download the ISO

This may seem obvious - but just in case you haven't already downloaded your Linux distro, make sure you grab the right one for your machine (if it's a new Mac, it's going to be the 64-bit ISO image - the filename usually ends amd64.iso. If you are downloading Ubuntu, you can get it here:

www.ubuntu.com/download/desktop

Step 2: Erase / Format / Initialise your USB Stick

It's time to pop your USB stick in your Mac. Regardless of whether you have a

How to mount a USB stick as a non-root user with write permission

So you want to use a USB stick or a USB hard drive, and you don't want to mount it as root every time?

Why would you want to do this?

- It's a hassle to mount the USB stick using sudo every time - you have to type the root password, and you have to specify all the mount options each time you mount it.
- The permissions on a FAT32 USB stick or drive don't allow write permissions as you, only root, so you have to sudo any write based file operation on the USB device. This is because the commonly found format of most USB disks is FAT32, as it has the best compatibility with Windows machines and is supported on Mac OS and Linux too, unfortunately FAT32 has no notion of file permissions, unlike EXT4).

Another point worthy of note, if you are someone who uses a shiny, full fat desktop like GNOME or KDE, then you'll likely find that things like USB removable media are automatically mounted for you, so it is often fine to use it in that capacity. However, for Linux luddites and server admins like me, that just won't cut it, I want an easy user-level mount at the command line, or bust!

Ok, you've made up your mind that you want to be able to mount the disk as a real user, not root. The good thing is, is that this is really quick to do.

Firstly, we need to create a directory where the drive is going to be mounted to. So if you haven't done so already, create this directory:

```
$sudo mkdir -p /media/<username>/usb
$sudo chown <username> /media/<username>/usb
$sudo chmod 0777 /media/<username>/usb
```

Obviously, you can call the mount point whatever you want, I've just called it `usb` and stuck it in the `/media/<username>/usb` directory (note `<username>` is your own username). The last two commands change the ownership to be 'you', and sets the permissions to read and write for all users (change this if you want it to be less open).

Next up, we need to ascertain which drive you want to mount. You may already know this, but if you don't know what the `/dev/` entry for the USB stick is, then you can find this out by sticking it in the usb port on your machine and running `dmesg`:

```
$dmesg

usb-storage 1-2.2:1.0: USB Mass Storage device detected
scsi host5: usb-storage 1-2.2:1.0
scsi 5:0:0:0: Direct-Access USB DISK 2.0 PMAP PQ: 0 ANSI:
6
sd 5:0:0:0: Attached scsi generic sg2 type 0
sd 5:0:0:0: 15133248 512-byte logical blocks: (7.74
GB/7.21 GiB)
sd 5:0:0:0: Write Protect is off
sd 5:0:0:0: Mode Sense: 23 00 00 00
sd 5:0:0:0: No Caching mode page found
sd 5:0:0:0: Assuming drive cache: write through
<strong> sdc: sdc1</strong>
sd 5:0:0:0: Attached SCSI removable disk
```

You can see that it's detected that a USB storage device on `sdc`, and that the partition it can see is called `sdc1` (highlighted above in bold).

Now all you need to do is edit the `/etc/fstab` file and add this line at the bottom:

```
<code>$sudo vim /etc/fstab

/dev/sdc1 /home/storage auto user,umask=000,utf8,noauto 0 0

</code>
```

Remove the 'noauto' bit if you want it to be mounted automatically on boot, however that's probably a bad idea if its a removable device!

That's all the hard work done, and you shouldn't have to do that ever again, now all you need to do to mount it, whenever you like, is to issue a simple mount command:

```
<code>$mount /dev/sdc1</code>
```

or, alternatively you can mount it by its mount point. Either way, it doesn't matter :)

```
<code>$mount /media/<username>/usb</code>
```

Hope this saves you some aggravation!

disk image. You can copy this to the clipboard to help with the next step if you like.

Convert images at the command line with ImageMagick

ImageMagick is a very mature tool. It's been around for donkeys years and it even acts as the silent 'back end' to some of the best GUI-based image manipulation software. However, the jewel in ImageMagick's crown is the tool called 'convert'. As you can imagine, this tool converts images at the command line. It can do so in many ways, for example, it can resize, change image quality, change formats (eg PNG to JPEG) and much, much more.

This is a great example of the power of convert from <http://climagic.org>: Here you can see how to make thumbnails of images with filenames in the range IMG_3000.JPG - IMG_3499.JPG:

```
for i in IMG_3*.JPG ; do convert -quality 60 -geometry 300 $i  
thumbs/$i ; done
```

The example shows that through simple one line command you can batch process many items with ease. Doing that with even the most substantial of image editing software is sometimes either impossible or a real challenge.

Here is one last example of its simplicity:

```
convert -resize 1024x768 original.JPG new.JPG
```

As you would expect, it simply changes the size of original.jpg down to 1024x768 pixels and outputs the new image in a new file called new.JPG.

What is X, XFree, XOrg or X Windows?

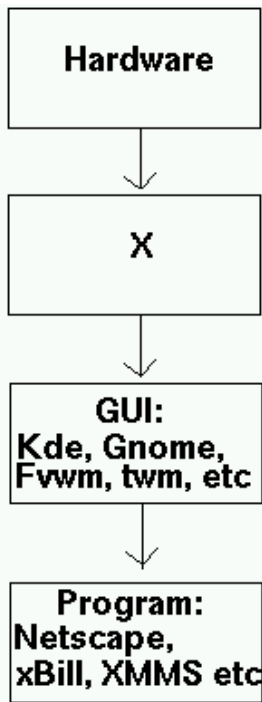


What X is Not:

- X is not a product by Microsoft with the letter X before it.
- X does not affect the way graphical windows on your screen look
- X won't let you browse through files in a graphical manager
- X is NOT a GUI (Graphical User Interface)
- X doesn't sound like it does a lot, does it? **Well read on to find out why it's an essential part of Linux!**
-

What X does:

X11 (and its variants, XOrg, XFree86 etc) is the layer between the hardware on your system (your graphics card, and so on) and the GUI that sits on top of X. Have a look at the following diagram to get the general idea:



When a program is started up, it goes through the process of first talking to the GUI, about what to do with it's windows, ie: placement, focus and so on. The GUI applies it's thoughts to the process, applies the look, menubars (File, Help, Close, Minimise, etc), and all decorations to the window, then passes it to X. X has the final decision on where it places it on a screen. It then talks to the hardware, making it issue the process.



Tux says: GUI stands for Graphical User Interface, and comprises of things like icons, menus, pointers and windows. It is usually pronounced 'Gooey'.

Without X in the equation, the GUI that you use, or properly termed: the Window Manager couldn't do anything.

X Is also a server. We usually refer to it as XOrg or just as X, but indeed it's proper title is an X Window Server. When you run X on your own linux box, X has to determine whether you want to display it on a local machine and on a local monitor, or on a remote monitor. You can also have more than one X session running on your linux box at the same time. If you have 2 monitors, and a dual headed graphics card (ie: Matrox G400 DH), you can get X to display Jane's X session on monitor 1 and Bob's X Session on monitor 2, all running on one computer, at the same time. Show me Windows Vista do that, please :)

Some X Heritage

If you're interested in knowing how X came to be, this chapter is for you.

X was an idea conceived by many: In 1984, Apple had released its first graphical user interface. If you were either too young, or never saw it, I reckon you should definitely take a look at the 1984 TV advertisement of the first Macintosh. It's a wonderful piece of art. You can see this great piece in computing history [HERE](#), although be weary- it's around a 13MB quicktime movie-- you'll need a fast connection if you want to see it in the next hour!

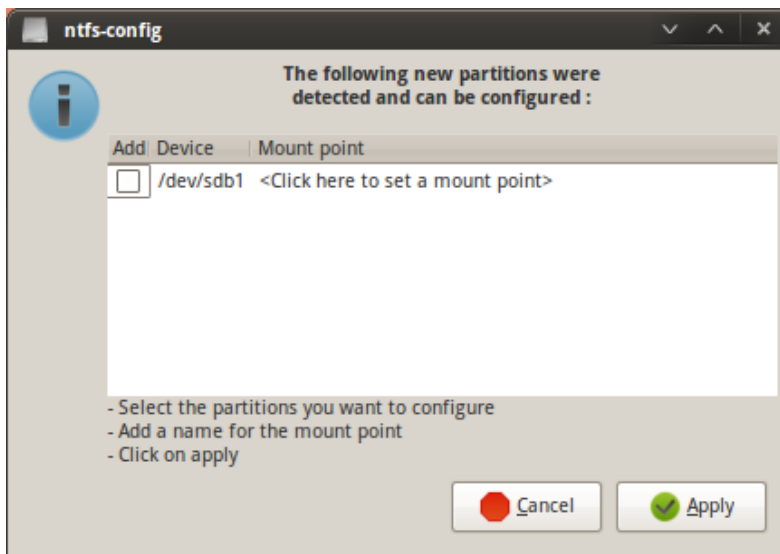
Even prior to this, a group of real boffins with some way far-out ideas (circa 1978) had been stroking their beards, drinking lots of hi-caffeine coffee and creating the very first GUI. It was made by the researchers in Xerox's PARC (Palo Alto Research Center), where they designed a GUI called Exlir, and terminal systems to go with it (called the Star). Unfortunately, it was not a commercial success, mainly due to high pricing.

Anyways... After Elixir, a few GUIs appeared (amongst others): Digital's GEM, Atari's TOS and the most famous of all, The Mac System OS, in 1984. Less than two years later in 1986, a consortium of UNIX developers including Sun Microsystems, Silicon Graphics and AT&T created X. Originally it shipped with the window manager called TWM. The source of this product and who created it, are still unknown.

TWM (perhaps, Tom's Window Manager, or, as most know it today; "THE Window Manager"), was the interface that sat on top of X, and allowed you to open up Xterms, resize windows and do basic window manipulation. It was pretty darn basic to say the least. Then came FVWM, which was a major improvement. For Linux geeks and for people who want a really light window manager, you'll find twm and fvwm still in use today. It still ships with all the latest distros. FVWM2 came out years later, along with other window managers such as NeXt/NeXtStep, AfterStep, IceWM, QvWM (which is supposed to look identical to Windows 98 btw), KDE, Gnome and many others. In 1990, Microsoft Windows 3.0 was released.

How to automatically make your Windows drives become available to Linux on startup.

Here's a handy utility that will save you any technical nastiness. It'll make your Windows drives (partitions) show up in the Ubuntu file manager so you don't have to mount them by hand each time you want to access them.



How to use the ntfsmount tool to automatically connect your Windows partition on startup

1. Launch the Synaptic Package Manager or the Ubuntu Software Centre.
2. Search for '**NTFS Configuration Tool**'
3. Install it.
4. Once installed make sure any drive you wish to Auto-mount is unmounted. Do this either by restarting Ubuntu and not mounting the drives, or by ejecting them via nautilus' side-pane or the terminal.
5. Launch the 'NTFS Configuration Tool' from System > Administration > NTFS

Configuration Tool.

6. In the NTFS Config. window, **check the box** next to the drive(s) you wish to automount.

7. Click Apply.

A new window will appear, however only check 'Enable Support For Internal Device.' . Your windows NTFS partition should now automatically mount on reboot each time.

How to read and write to Windows NTFS drives as any user

So you have a Windows hard drive using the NTFS partition type. The good news is that most Linux distributions these days can read and write to it automatically without as much as a config change. It automatically sees the partition and mounts it

That's great, but what if you have a one-user Linux box and you want every app on your Linux box to be able to use the files on that partition, not just your user account? I recently ran into this problem when I wanted to share an external NTFS formatted USB drive with my Plex Media Server. The Plex media server runs as the pseudo user 'plex'. My own user account obviously is not named 'plex' and therefore it refused to see any of my music and movies on the USB hard drive, as the files on the hard drive were 'owned' by my user account, not 'plex'.

How to get around this problem? Well, provided you are not worried about sharing everything on that drive with all (or some) of the other real or pseudo users on the machine, then you can create a user group, say called 'ntfs', and have all the users you want to read and write to the drive in that group. Here's how you do it from the command line:

```
sudo groupadd ntfs

sudo usermod -a -G ntfs YOUR_USER_NAME

sudo usermod -a -G ntfs
USER_NAME_OF_ANY_OTHER_USER_YOU_WANT_TO_ACCESS_THE_DRIVE
```

So the above has created the group 'ntfs' and added your own username as well as any others you want to that group. The output of the first command should look like this:

The output should look something like this:

```
Adding group `ntfs' (1004)...  
Done
```

Take note of that number in brackets. That's your GID (group ID number).

Next, let's make the location to mount the partition on your drive. Assuming you are running Ubuntu, this will be in /media, but it can be anywhere you like, for example /mnt, or even under / - just as long as all the users you added to the group can already access that folder.

```
sudo mkdir /media/windows  
sudo chgrp ntfs /media/windows
```

Now it's time to edit the file system table (fstab). Don't worry - that's not as scary as it sounds, it's just a text file which contains a list of the partitions the Linux system should mount on startup.

```
sudo nano -w /etc/fstab
```

Assuming that Windows is installed on the first drive, and first partition we use /dev/sda1. If your windows drive is on another drive in your PC, say the second drive, and it's the third partition, it would be /dev/sdb3 and so on. You can check to see if you got the right drive and partition number with the fdisk tool.

add the following to the bottom of the /etc/fstab file:

```
/dev/sda1    /media/windows    ntfs-3g  
auto,gid=1004,unmask=0002    0 0
```

Remember to keep the spaces after each item as they instruct the system to read each option. Don't forget those two zeroes at the end of the line either!

Explanation: /media/windows is the new location where the partition is mounted, so when you visit it in your file browser (or with ls at the command line), you'll see the files in /media/windows. The option *ntfs-3g* is telling the mount program that this is a ntfs partition and we will use the 3g driver to write to it. The next option tells the system to mount the partition automatically at startup and finally the gid/umask information allows all users in the ntfs group to read and write to it. Note we specified the gid of 1004 which is the gid we were given by the groupadd command. If you don't match this number, you and any other user in the newly created ntfs group won't be able to read and write to the /media/windows folder.



Make sure that the `gid=` value is the same as whatever you saw when you used the `groupadd` command earlier.

Save the `fstab` file and exit the editor. To test it works properly, simply type:

```
sudo mount -a
```

This command reads the contents of the newly updated `fstab` and as long as it is correct, it will mount the windows partition in `/media/windows` (or wherever you specified to mount it). When you reboot your machine the partition should automatically be mounted so you shouldn't need to do anything!

Enjoy!

How to use a Mac to create a Linux Live USB Stick and Boot it

Ok, let's say you are just about to take the plunge and install Linux on your nice shiny Mac but before then you want to test drive how it really works, using a Live distro. There's a few ways to boot a mac up, firstly there's the good old CD ROM drive (SuperDrive as you mac fans know it), but guess what? Most of the new MacBook Pro's etc don't even have a DVD/CD ROM drive! So scratch that... Next there is network boot and finally there's boot from USB. Network boot is a whole other ball game that we will cover separately for good reason so let's concentrate on how to create a USB stick which is bootable and contains a Linux Operating system of your choice on it. The good thing about this tip is that it's quick and easy. The other, official ways of booting an Apple/EFI system can be quite complex, including Ubuntu's own solution.

I tested this on a Macbook Pro Retina (Late 2013 model), however this should work on any modern Mac with EFI booting. Note that the tool seems to have been tested with Ubuntu and it's derivatives, if you are trying another distribution like RedHat/CentOS, this may not work for you.

What you'll need

- 1) A Mac with EFI ROM running Snow Leopard or Later ([How do I tell I am running an EFI mac?](#)).
- 2) You'll need the easy to use Mac - Linux USB Loader tool from [this web site](#) (SevenBits).
- 3) An ISO download of your Linux distribution of choice. Be sure to chose a normal 64-bit version made for PCs, don't download an Apple specific version as we require the distribution to support EFI booting. EFI has replaced the good old BIOS in the new macs as well as many newer PCs.

Step 1 - Format your USB stick

Pop your USB stick in and fire up Disk Utility (a built-in App on your Mac). You will need to format the USB stick as FAT32 (MS-DOS FAT) format, and it must have an MBR (Master Boot Record). Make sure that there is one partition on the USB stick and format it (this will destroy any data on it).

Step 2 - Copy the ISO image to your USB stick

To do this, simply fire up the Mac-Linux USB Loader tool as listed in the 'What you'll need' section of this article.

Click on 'Create Live USB'. Select the appropriate ISO image from your Finder and sit back for a while.

Step 3 - Boot Linux

All that's left to do is restart your Mac and boot Linux from the USB stick.

To do this, shut down your mac, turn it back on. When you hear the usual Apple 'Chime', press and hold the Option key. You will see all your drives including the inserted USB stick which will be yellow in colour and will probably be entitled 'EFI Boot'.

Click on the yellow icon and you will see a text screen loading the Linux kernel if all went well.

How to install Linux on a Macintosh computer

So, you've got one of those shiny Macbook Pro retinas?

..But you are a Linux fiend, and now you want to rid yourself of the poor mans' Unix that Apple call Mac OS X? This tutorial is for you!

This is an advanced tutorial which works at the command line and can cause irreparable damage to your data. If you are a novice, it is not recommended that you undertake this tutorial. If you do proceed, make sure you have backed everything up with TimeMachine or such like tools. I must also acknowledge the awesome work of Jessie Frazelle, her blog entitled 'Linux or Death' was the inspiration for this article, and much of it is shamelessly cribbed from there, simply because it worked, unlike any other blog on the subject we found on the interwebs!

This tutorial has been tested on a late 2013 Macbook Pro 15", however it should work with any EFI based Mac (more on that in a bit). EFI based Mac's started around 2008 (you can check the list of the Apple EFI systems [here](#)). This should include Macbook Pros, Macbook Air, iMac etc...

I am also working on the basis that you want to keep Mac OS X on your hard drive and that you wish to dual-boot it at any time. You should have plenty of free space on your disk drive (the more the better), so either delete some cruft or move some of your old data onto a separate external archive hard drive (because I know you got one or ten of them lying around!).

Finally, we used Mac OS X 10.11.1, 'El Capitan', which is the latest OS X at the

time of the release. El Capitan introduced a 'security feature' called 'SIP' (System Integrity Protection) which you will additionally have to overcome if you are using El Capitan or newer. More on that in a bit. We will be installing Ubuntu, specifically Ubuntu GNOME 15.10, but this should apply to any Linux distro more or less, although your mileage may vary with Video stuff particularly.

The tutorial you are about to read has six main sections. These are:

- Installing the EFI boot manager
- Downloading and converting your Linux distro of choice
- Partitioning your hard drive
- Installing Linux
- Finishing up and nice to have items.

Installing the EFI boot manager

EFI stands for Extensible Firmware Interface and is now pretty much commonplace in Macs and PCs across the industry. It replaced the trusty old BIOS system that PCs had used since the 1980s. Installing Linux on a BIOS based machine was trivial, but now with Apple's take on EFI on their customised hardware, it can be a little challenging. No worries, this is the Ultimate Linux Newbie Guide. We got this!

Disabling SIP

Before we go ahead and install rEFInd, we will need to take care of that pesky SIP (System Integrity Protection) rubbish. There are a couple of ways to do this, but we found the easiest way to do so, is pop your system into recovery mode and issue a command from the terminal there. There is a bit more information on this process over [here](#).

To enter recovery mode on your Mac, shut your machine down completely. Give the machine around 30 seconds and then switch back on. Now quickly hold down the Command and R key at the same time until at least you hear the Apple 'chime' sound. Shortly you will enter recovery mode. I recommend plugging in an Ethernet cable to do this, however it is possible to do with WiFi.

Once you are in the Recovery tool, enter the Utilities menu up on the top bar, and click on Terminal.

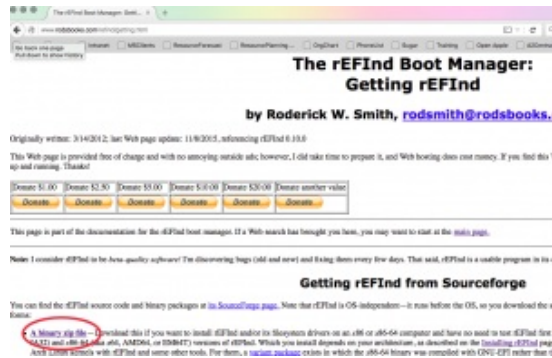
Issue the following command:

```
csrutil disable
```

That's it! Restart your computer and you are done. If you ever want to re-enable the SIP functionality (why?), then you can transpose `disable` with `enable` using the same steps.

Now you should head back to the terminal. We are ready to install `refind`!

Download `rEFInd`



The red circle indicates where to download `rEFInd`

`rEFInd` is a boot-loader for EFI based machines. Think of it like bootcamp, or GRUB for GRUB :) You'll want to download `rEFInd` from the `rEFInd` website:

- [rEFInd Website](#)

Now, if you take a look around the `rEFInd` website, you'll see it looks like the guy that wrote it believes in punishing everyone that wants to use it. It took us about 20 minutes just to find the frigging download link! So the ULNG has taken the time to go through all the pertinent steps to make it shit tons easier for you!

The version of `rEFInd` that we used is 0.10.0, and we used the [zip archive version](#). Once you download the binary, you are going to need to start the rest of your work from the Terminal, so open up the Terminal from the Utilities folder on your mac and head over to your Downloads folder where you saved `rEFInd` to.

If the zip archive is not already unzipped, unzip it using the `unzip` command and head into the newly created `refind-bin-0.10.0` folder:

```
$unzip refind-bin-0.10.0.zip  
  
$cd refind-bin-0.10.0
```

We now want to install `rEFInd`, and we should install all the EFI drivers just in case we need them at any time. `rEFInd` 0.10.0 offers a substantially more improved installer than previous versions. Which is nice :)

```
$sudo ./refind-install --alldrivers
```

Now it's time to edit the EFI config file, but you will need to mount that hidden EFI partition first. Thankfully, rEFInd has a little tool you can use to mount the partition:

```
$sudo mountesp
```

Edit /Volumes/ESP/EFI/refind/refind.conf. Like us, you may find the refind.conf file is in /Volumes/ESP/EFI/BOOT, instead of a folder called refind. This is probably because we fiddled around with rEFInd and its predecessor, rEFIt before. Just because.

```
$vi refind.conf (or nano, if you are that way inclined. Just not emacs!).
```

locate the line that says scanfor and edit it to say:

```
scanfor internal
```

If no such line exists, add it into the file near the top.

Load the linux file system driver. Check for a line that starts fs0. If no such line exists, add it as below, otherwise edit it:

```
fs0: load ext4_x64.efi
```

```
fs0: map -r
```

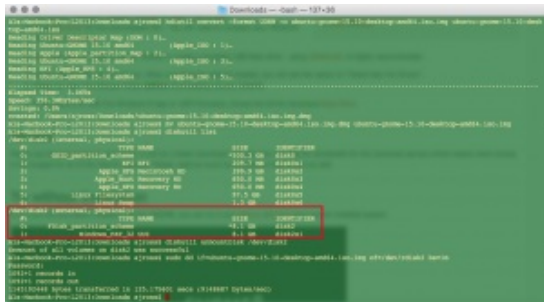
Save the file and quit your editor. That's pretty much it for the rEFInd bit. That is the hardest part over and done with. If you want to be sure it worked, you should power off your machine and power on again. If you see a grey screen with the rEFInd logo, then it has worked. You should be able to chose the Mac OS X logo and hit return to start up OS X again.

Downloading and converting your Linux distro image of choice

Next, unless you haven't already downloaded the iso image of your choice, it's time to go grab it. For this particular demonstration, we are using Ubuntu 15.10, however most other distros should work. Using more hard-ass systems like Arch or Slackware, or even Debian, will be more challenging. This is challenging enough, so

do what you will, but we will stick to the easier distros for now!

Make sure you download the x64 version of the distribution you choose, if there is an EFI boot version, choose that also.



Next, you need to convert the iso file into an image file that Mac OS X recognizes so that you can stick it on a USB stick to boot from. Although you are back at the terminal for this, thankfully it's not that onerous a task.

```
cd (directory where you downloaded the distro to)
```

```
hdiutil convert -format UDRW -o <<filename you want.iso.img>> <<filename of the download.iso>>
```

(so, in other words, *hdiutil convert -format UDRW -o ubuntu1510.iso.img ubuntu-gnome-15.10-desktop-amd64.iso* would convert the latter file into *ubuntu1510.iso.img*, which would be in the RAW, or more specifically UDRW format).

Mac OS likes to rename the file with a .dmg extension, to avoid confusion over that, it's probably best that you rename it. A simple mv command will sort that one:

```
mv ubuntu1510.iso.img.dmg ubuntu1510.iso.img
```

(renames the file *ubuntu1510.iso.img.dmg* to *ubuntu1510.iso.img*).

Next, quickly type in *diskutil list* in the terminal. It should list one disk drive (if you have any external drives connected, disconnect them for now). Take a note of your Mac OS disk. It should be called */dev/disk0*. Look for tell tale signs in there like 'Macintosh HD'

Next, locate that USB stick of yours and whap it into the usb port. Please note that you want to have an empty USB stick, because this process *will* destroy any existing data on it.

Once you have the USB stick plugged in and all settled down, issue the following command:

```
diskutil list
```

You should see a list of disks like the one in the screenshot to the above. Note the red box around `/dev/disk2`. You can see that it has a Windows FAT32 partition, our usb stick is 8 GB big, you can see the size there is 8.1GB, which is clearly much smaller than the main hard drive, which is 500GB in size. Just make sure you identify which disk your usb stick is and take a note of that. The one we are using is `/dev/disk2`.

Now you need to unmount the usb disk, to do that issue the following command

```
diskutil unmountDisk /dev/disk2
```

Obviously if your USB stick is not `disk2` then change it to suit. It should tell you that the unmount of all volumes on the disk were successful. Now all we have to do is copy the UDRW version of the iso image to your USB stick. For that, we will use trusty old `dd` :)

```
sudo dd if=ubuntu1510.iso.img of=/dev/rdisk2 bs=1m
```

The things you have to be aware of here:

1. This is a fully destructive command. It will blow up any disk you point it to, that's why we made sure you got the note of that USB stick in `diskutil list`, above.
2. `if=` is short for input file, meaning the input file is the name of the iso file you converted into the `iso.img` file. We are using the shortened filename here just for an example filename.
3. `of=` is short for output file. The output file, is in fact a device. In this case, we are using `(r)disk2`; `disk2` obviously is the drive we noted earlier, we use the `r` in front of it because that allows raw device access, so its marginally quicker.
4. `bs=1m` means block size, 1 megabyte. You don't have to use this option, however it will significantly increase the speed of the transfer to your USB stick.

Mac OS is going to bitch and moan that you have a volume it can't read inserted. If it says Eject, click that, otherwise type `diskutil eject /dev/disk2`.

Partitioning your hard drive

This part chops your disk up in the way you want it. Back in the bad old days of Linux, you used to have to download a partition manager and use that to resize your disk. These days, that's mainly gone, but because of Journaled, Encrypted HFS+ partitions, that's not a good idea to let a Linux based partition manager loose with, unless you want to toast your Mac OS data.



Using Disk Utility to resize your Mac OS X hard drive partition for Linux

Thankfully, this can easily be solved. In your Utilities apps folder, you'll find Disk Utility. If you like, quickly scan your hard drive for errors, just to make sure it's all toot sweet before we get down to business. Repair any errors you may have.

Once you are ready, you will see a list of internal drives on the left hand side. Your Disk Utility may look different if you are using an older version of OS X, but it still offers the ability to resize a volume.

On the hard drive that your Mac OS X partition exists on, click on the top drive, not any subsequent partitions listed below it. Click on the 'partition' button, and you'll see a pie chart like the one above (don't worry if it's not quite like that). You will see you can move the slider around the pie to resize your partition(s). Pull the size slider back for the Mac OS partition to release the free space on the disk. Make a blank partition until you have enough space for your new Linux system. As much space as you are willing to.

Apply the changes and let the resize operation complete. If you have an SSD, this should be relatively quick (a few minutes).

Installing Linux

Woo-hoo! This is the fun part! Now we get to install the operating system that your mac has been longing for!

Switch your mac off completely. Connect your Ethernet to Thunderbolt adapter

and your USB drive we made earlier. If you don't have one of those thunderbolt adapters, life is going to be tricky for you, you are going to have to download the wireless drivers and install them manually to get things working. If you don't have one of the adapters, ask a friend for one, or buy one cheap from Ebay or such like. It will save your sanity.

Turn on your computer and hold down the option/alt key. You'll see a menu pop up which you can see your Macintosh HD as well as the USB stick (it's a yellow looking drive thing). It will be named EFI or something similar. Use the cursor keys to select that and hit return. PS: if you are having issues with your bluetooth keyboard at this point, make sure you revert to using your laptop's keyboard and mouse for the time being.

At the step where you have to choose the partition and you are using the Ubuntu style installer, select 'Something else' from the options around partitioning. Locate the empty partition you made and create 2 partitions out of it. Make a big partition and a small partition (roughly 8-16 GB in size). The big partition should be the remainder of the free space. The big partition should be ext4 in type, and should be formatted with the mount point of "/". The small partition should be formatted as swap.

Press next and let the good times roll. Everything else should be pretty normal.

Once the install has finished, restart the computer, no need to hold down alt/option this time. All going well, you should be seeing the rEFInd menu. Use the cursor key to select your Linux installation and hit that return key. Fingers crossed, your system will start up without much of a hitch!

Notably though, you probably won't have a few things that work out of the box. Most of these can be covered off on the next section.

Finishing up and nice to have items.

- Wifi.

Some lucky people will experience that their WiFi works straight out of the box. For most of us, we are going to need to install it. You can do this with:

```
sudo apt-get install firmware-linux-nonfree broadcom-sta-  
dkms
```

The broadcom wl, bcwl43 and fwcuttler software should all be installed. Reboot to

enable the puppy and all should be good when you select your wifi from your system's networking setup menu.

- Graphics.

The graphics display should generally work out of the box, however there may be 'interesting' graphical issues. Not all of these might be fixable, but give the NVidia drivers a try, and if you still don't have any luck, read the many forums until you get a solution that works for you.

```
sudo apt-get install nvidia-driver xserver-xorg-video-intel
```

Note if you are not using xorg, you'll need to make the appropriate changes here. Maybe best to stick with xorg for now!

- Screen backlight.

This is a /bit/ of a hack, but it works!

Check out [Jessie's blog](#) for the screen-backlight script. The screen backlight section is near the bottom of the blog article.

- Keyboard backlight.

Again, Jessie Frazelle to the rescue here. The section below the screen-backlight section has a [keyboard backlight](#) bit too!

- Mounting your MacOS X partition.

Okay cokey. Now here's the thing. Apple can be real pains in the asses some times. It is quite likely that you have what's called CoreStorage, if you have anything OS X 10.10 or newer. This provides an encrypted, journaled file system; even if you haven't installed FileVault2 (if you have, turn that off!).

To give full read/write access to your Mac OS X partition from Linux, you will need to revert it back to standard HFS+. To do this, you can pretty much enter one simple non-destructive command.

First up, at the terminal, issue the command **diskutil cs list**. You will see something like the below. If you know LVM in Linux, this is pretty much the same thing. Your main Mac OS X partition (Logical Volume) should be in Apple_HFS format.

```

CoreStorage logical volume groups (1 found)
|
+-- Logical Volume Group 56EDEE99-57E6-495B-A809-7042CBB9F725
=====
Name:      Yosemite
Status:    Online
Size:      39349997568 B (39.3 GB)
Free Space: 259604480 B (259.6 MB)
|
+--< Physical Volume 3D2DC4ED-4F33-4808-92F7-BCD6BFC36CA2
|-----
| Index:    0
| Disk:     disk0s4
| Status:   Online
| Size:     39349997568 B (39.3 GB)
|
+--> Logical Volume Family 993EBF39-FE2E-4F2C-BD44-76E460B9EC7A
-----
Encryption Status:  Unlocked
Encryption Type:     None
Conversion Status:   NoConversion
Conversion Direction: -none-
Has Encrypted Extents: No
Fully Secure:        No
Passphrase Required: No
|
+--> Logical Volume 47F9D6B1-F8F2-4E64-8AD4-9F2E2BD78E29
-----
Disk:      disk2
Status:    Online
Size (Total): 38754844672 B (38.8 GB)
Conversion Progress: -none-
Revertible: Yes (no decryption required)
LV Name:    Yosemite
Volume Name: Yosemite
Content Hint: Apple_HFS

```

As long as the 'Revertible' flag is set to Yes, you are good to go. Simply enter the following command:

```
diskutil coreStorage revert
```

The long string of stuff is that big long alphanumeric string of text highlighted in the red box, you want to use copy and paste it to make sure you don't make a mistake!

The conversion took ages for us, however your mileage may vary, depending upon how much data is on your drive, and how fast your drive is. If you type `diskutil cs list` again, you'll see how much % of the conversion has been accomplished. Don't reboot your machine until that's over and done with, but after then, you can safely mount your OS X partition with full read/write access.

First, make sure that you have hfsprogs installed. Example installation command:

```
sudo apt-get install hfsprogs
```

Next, mount or remount the HFS+ drive; commands need to be as follows:

```
sudo mount -t hfsplus -o force,rw /dev/sdXY  
/media/mntpoint
```

or

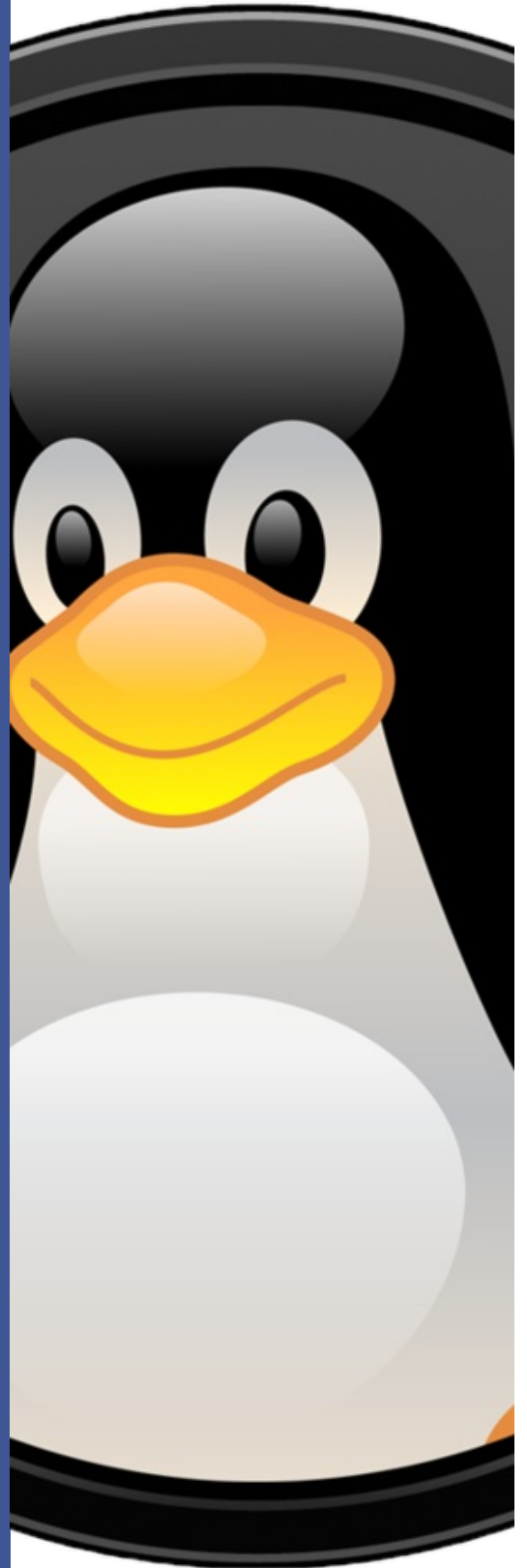
```
sudo mount -t hfsplus -o remount,force,rw  
/mount/point
```

Okay, that about wraps it up for this ditty, I hope it has worked for you. If it hasn't, or you have some feedback to offer, we would love to hear it! Drop it in the comments, y'all :)

SysAdmin Tips

The following pages have been selected from the website as an example of the more advanced topics that would be suitable for a Linux user who is growing their knowledge towards working with Linux as a career.

[Check out all of the tips on the website](#)



Command Line Interface

What's a Terminal/Console/Command Line/CLI?

In some parts of this site, you may see references to using the Console, Terminal or CLI. What this means is that you are required to use a text-mode command system, rather than a graphical command system to perform actions. In most Linux systems, the command line interface can be accessed by looking in your Applications menu and searching for 'Terminal'. Other apps such as Xterm, Rxtv, Konsole and others exist, but they all generally do the same thing.

So, why should I use a CLI?

Commands used to be the way that all computers were operated. Since the beginning of keyboards and monitors, computers required users to provide text based instructions to do things like work on files or use applications. A command interface is often abbreviated to CLI (Command Line Interface or Interpreter). The commands you give are usually entered using a piece of software called a Terminal or Console. This software is usually a small tool such as the Gnome Terminal, xterm, rxtv or the Linux Console which traditionally runs the Bash (Bourne Again Shell) in Linux. The commands you give to the bash shell are interpreted to provide outcomes, for example the command 'chmod u=rwx file.txt' without the parenthesis would change the ownership permissions of the file called file.txt to be readable, writable and executable by the owner of the file (user).

Today of course, most computers are synonymous with the use of a Graphical User Interface or GUI, which uses Windows, Icons, Menus and Mouse pointers (WIMP) to operate the computer rather than commands, however many systems administrators and power users in both Windows and Linux worlds still prefer to use a command line interface to quickly perform actions. It's a great time saver once mastered!



The Gnome Terminal Application can be found under the Applications > Accessories menu in the Debian Desktop.

If you want to know about some of the commands which you can type in at the Terminal, check out the [commands tutorial](#).

Files, Directories and the Linux Filing System

Tell me about /usr, /dev, /etc, /home and friends

You may have wondered what goes where on your Linux system, and what all of the directories or folders in the / (root) [folder](#) on your system are doing. This table lists a typical root partition on a standard Linux system. Yours may differ slightly, but the main ones (/bin, /usr, /lib, /var and /etc will always exist).

Folder Name	Contents
/bin	The rudimentary minimum set of Linux programs are held here. These programs are essential to the operation of your Linux system. It's like a system folder, except it only holds binary programs (as the name /bin suggests). Don't delete anything in here or your system will most likely become broken!
/usr	/usr is a place where binary programs are installed into. It is also where your kernel source is held. There are various folders within /usr, such as bin (where installed software goes), src (your Linux Kernel source is here), X11R6 or X11 , (The X Window system is here), share (things like icons and pictures, wallpapers and fonts are held here -- so that they are shared between your systems users), doc (where the programs on your system hold their documentation or manuals), lib (where all the libraries, such as Qt or an OpenGL driver would be held) , etc (global configurations for installed software) and finally local (local binaries and programs to your system are located here, just another place for bin really).
/dev	Large directory which holds files that link to real hardware devices attached to your system, ie: /dev/fd0 is the file for the first floppy disk drive (fd) on your PC. /dev/scd1 is the second SCSI cdrom drive attached to your PC and /dev/ttyS0 is the equivalent of COM1 in DOS. Due to this handy architecture, you can point the output of textfiles or running

	<p>programs straight to a hardware device, without actually knowing how the hardware works. The kernel knows that. I.e, doing a <code>cat /home/my_modem_at_instructions.txt > /dev/ttyS0</code> would send the contents of the file <code>my_modem_at_instructions.txt</code> to the modem on COM1 (or <code>ttyS0</code>).</p>
/etc	<p>An important directory that you should try not too much to fool with (although some times, you've gotta go there to change things). This directory can be basically described as a configuration directory for system-wide applications and resources. For example, the file <code>/etc/lilo.conf</code> holds the configuration for the LInuxLOader program -- it boots Linux from your hard drive for you. If you edit this file, this will allow you to change the way that your system boots. The file <code>/etc/X11/XF86Config</code> holds the basic configuration for the X server. Don't delete this folder. Your system will crumble!</p> <p>Note: individual users program profiles are stored in their home directories, i.e: a single users profile for the bash shell is stored in <code>~/.bashrc</code> (a file with a dot in front of it is a hidden file btw).</p>
/home	<p>This is where all users home directories should be. For example, if your username is <code>jdoe</code> then you will probably have a home directory called <code>/home/jdoe</code> or <code>/home/users/jdoe</code>. All of <code>jdoe</code>'s personal files (excluding applications generally), profiles and directories will be stored in here.</p>
/root	<p>The root user's (system administrator) home directory. Root and ONLY root should be able to access this folder.</p>
/lib	<p>Where standard system libraries are stored, such as the pam (Pluggable Authentication Module) library.</p>
/mnt	<p>On a UNIX system, all drives are mounted before and after use, instead of just being A:, C: and D:. This means that a drive can just be part of your filesystem (as a directory). Most of the time, people like to put their mounted drives (such as a CDROM or Floppy) in the <code>/mnt</code> (short for mount) folder. Typically, you might have the following folders within a <code>/mnt</code> directory: <code>cdrom</code>, <code>floppy</code>, <code>cdrw</code> and possibly a windows drive: <code>win_c</code>. Whatever you have in the <code>/mnt</code> directory, you will find that the file <code>/etc/fstab</code> (file system table) has an entry relating the <code>/mnt</code> directory to the physical drive. I.e: <code>/mnt/cdrom</code> usually points to <code>/dev/cdrom</code> in the <code>fstab</code>, and <code>/mnt/floppy</code> usually points to <code>/dev/fd0</code>.</p>
/proc	<p>If you have a good look at the <code>proc</code> folder, you will find that it's not actually a real folder at all, it's a virtual folder - it's 0 bytes big. Inside <code>/proc</code>, you will find handles for various devices and also, some informative files such as</p>

	meminfo (if you type <code>cat /proc/meminfo</code> it will tell you how much memory you have and are using). <code>/proc/pci</code> holds all the information about your PCI bus and cards attached to it. Have a look, you can't edit any of these files.
<code>/tmp</code>	Temporary files usually created by running software are held here, sometimes these are called sockets.
<code>/sbin</code>	Where system programs are placed. Much like <code>/usr/local/bin</code> and <code>/usr/bin</code> , however, these binaries are usually ran as root, for system maintenance reasons. For example, RedHat/Fedora systems store the <code>ifconfig</code> tool here (which will allow you to alter network settings)
<code>/var</code>	Holds files that are generally server oriented, such as mail spools (where mail is held for a user until they pick it up). Lock files are held in <code>/var/run</code> . These files stop multiple instances of one program running by one user if necessary. Red Hat 7 and above also puts the web server in this directory, although Apache's default directory is <code>/usr/local/apache</code> .
<code>/lost+found</code>	Damaged data that has been found on a drive when <code>e2fsck</code> (equivalent of a Scandisk in Windows) runs is put in this folder for personal examination by root.
<code>/opt</code>	Placed on your system generally when KDE is installed. Some KDE and Qt Based programs like to install themselves in the folder <code>/opt/kde/bin</code> for some reason.

Keeping your files nice and tidy

It's important that you keep your files tidy on your Linux system, mainly because you'll find them easier, but also because you want to keep your system secure. Place all of your own personal files in your own folder, and make sure that the permissions are set that only you can access them. For example, if you have lots of documents, consider making a folder in your home directory called `docs`. In `docs` you can put all of your documents.

Making folders

To make a folder, you can usually enter your file manager in Gnome or KDE and **right click** on a white space in the folder that you want to create the folder, then select Create Directory or New Folder.

However, to do it at the terminal, change to the directory that you want to make the folder in (`cd /name_of_folder_you_want_to/go_to`). Then type:

`mkdir new_folder`

Operating on files: Executing (running), copying, moving, renaming and deleting files

Binary Execution

To execute a binary program, at the terminal, you can do the following:

`/folder_of_program/name_of_binary_file`

So, if I wanted to execute the file `myscript.pl` in the current directory, I could issue:

`./myscript.pl`

Some programs such as LokiGames software (www.lokigames.com), Sun's *StarOffice* and Corel's *WordPerfect Office* have their own dedicated binary installers. Most of the time, these sort of binary installers are initially executed from the terminal, by using the example above.

Note that sometimes installer packages come without permission to execute (run), so you will need to provide it permission, try `chmod 755` before you run the program.

Copying files

To copy a file at the terminal, the `cp` command is used, with the following syntax:

`cp source_dir/source.file destination_dir/destination.file`

For example, I want to copy `myfile.pl` from the current directory to `/home/bob/perl`:

`cp myfile.pl /home/bob/perl`

Moving files

To **move files** at the terminal, the format is very similar to the `cp` command. The move command is `mv`.

Here is the standard syntax:

`mv /source_dir/source.file dest_dir/source.file`

So, as a typical example:

`mv myscript.pl /home/bob`

The mv command has many other arguments (options you can pass to it at run time), such as -R, lets investigate the -R switch:

```
mv /home/jane/docs/* /home/bob/docs -R
```

This would copy the entire contents of the folder /home/jane/docs (because we gave it the wildcard *), to the folder /home/bob/docs. Additionally, as we passed the -R argument, if there were any further directories inside /home/jane/docs (ie: folders called CV, letters and notes could exist inside the docs folder), the -R argument moves the CV, letters and notes folders as well as the docs folder.

Renaming files

Unlike DOS, which had a separate command to [rename files](#) with (ren), Unix simply uses the move (mv) command to do it. Take this example:

```
mv oldname newname
```

That would rename the file the file in the current directory called oldname to newname.

Deleting files

[Deleting files](#) can be done simply with the rm command. Deletion of empty directories can be done with rmdir command. Here are some examples:

Deletion of a [single file](#):

```
rm myfile.txt
```

Deletion of every file in current directory:

```
rm *
```

Deletion of every file in current directory and all directories within it (recursive deletion):

```
rm * -Rvf
```

Note: if you want to delete a directory and all of it's contents in one swoop, instead of using rm *, then cd .., then rmdir dirname, you can simply do a **rm dirname -Rvf**.

Removal of an empty directory:

```
rmdir mydir
```

Security and files: Permissions and Ownership

In Unix, all files have permissions. Enabling files to have permissions allows security over and above simple password protection to a system. Imagine the situation: You have a Linux server which is used for 5 people from your companies Research and Development department. The five people are:

- bob
- jane
- mary
- mark
- simon

Consider for a moment that Bob wanted to keep his data (which is stored on the linux server over the network) private from jane, mary, mark and simon. Without the use of permissions, it would be impossible to do this. So for this reason we have the following permission system in Unix:

```
/home/bob    bob  users    drwxr-xr-x
```

What you see above is an edited example of a directory listing on a unix system. You can achieve the same result as what you see above by using the ls command with the long listing function (ls -l). Consider that the above result was an abstract of the output of ls -l /home on the linux server.

The part that reads rwxr-xr-x is the permissions bit for the file (or directory). The bit that says **d**, can also read **-**, **s**, **l** or **c** - means normal file, s means sticky, l means symbolic link (like a shortcut, but but better) or c is a block device (you'll find these in the /dev folder). The **d** bit that is on the listing that we have is because the listing is of a directory.

So, from the above listing already we know that it is a directory, it has some permissions set, and that the directory is called bob and it is in the /home folder.

The Permissions for /home/bob

But what do the permissions rwxr-xr-x actually mean, I hear you cry?! Have a look at this:

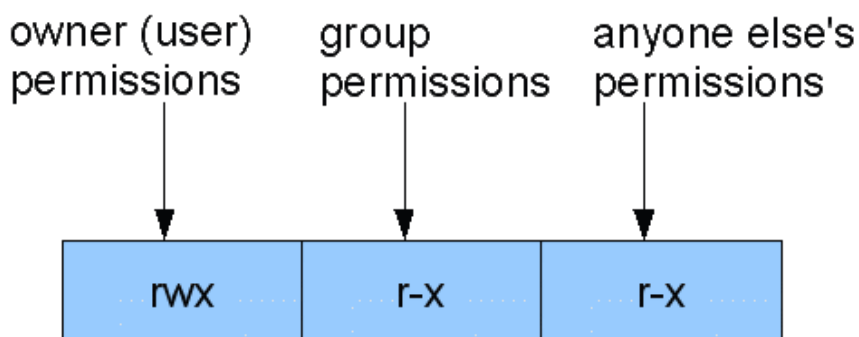


Figure 1: Explanation of permissions on a Linux system: UGO = Users, Groups, Others.

As you can see from the diagram, the first rwx is mapped to the owner of the file. This means that the owner (which is presumably bob), has **read (r)**, **write (w)** and **execution (x)** access for that folder.

The second permission, r-x is mapped to the group of the directory. Not only can an individual user own a file, but a whole group (if desired) can access a file. This example illustrates that everyone in the group users can read the files in the directory, and that they can execute programs within it, or change into that directory.

The third permission, r-x, is for others (commonly referred to as world), which means everyone else. This is the most dangerous permission bit. Our example allows anyone on the server to be able (like the users group) to read anything within the bob directory, and execute files in the directory.

Ideally, what bob should really do, is set the following permissions, for total privacy:

```
/home/bob    bob  users    drwx-----
```

Using the chmod command to set permissions

When you want to change the permissions of a file, you can use the ch commands: chmod (which changes the rwx bits of a file), chown (which changes the owner of a file or directory) and chgrp (which changes the group which accesses a file or directory).

This part deals with chmod.

Imagine a file called cv.doc with the following permissions User: read, write, execute. Group: read. Others: none, ie:

```
cv.doc rwx/r--/---
```

Let's imagine that we wanted to change the permissions to: rwxrwx---, so user and group can read, write and execute the file cv.doc.

This command would let us change the desired permissions:

```
chmod ug=rwx,o= cv.doc
```

So, what we did there, is we ran the chmod command and told it to set the **'user'** and **'group'** permissions to read, write and execute. We set the **'other'** (everyone else) permissions to nothing. Finally, we specified the file that we wanted to change the permissions for, cv.doc. Take this other example, as with Unix, there is *always* more than one way to do something:

```
chmod 0770 cv.doc
```


You will notice the only difference to the above command is the 0770 against the ugo bit. The 0770 (or just 770 would do) means exactly the same thing as `ug=rwx,o=`. 0770 is the numerical representation for the permissions, octal is a base 8 number system, and the permissions of files are based on this. ie:

- 1 = execute only
- 2 = write only
- 4 = read only

You will notice, there are 8 (as in octet) combinations of numbers (0 through 7). If you add 1, 2 and 4 together, you get 7, which is read, write and execute. By taking any combination of all of the numbers 1, 2 or 4, you get your desired permission that you require. For example, $1+2 = 3$, and `chmod 300` would give you write and execute permissions.

It really doesn't matter which way you choose to use `chmod` (either ie: `u=rwx,g=rwx,o=rx`) or `(775)`, just use what you feel most comfortable with, although it is handy to know how each way works.

Using the `chown` command to set the owner of files

The syntax of the `chown` command is relatively simple:

```
chown bob mydoc.txt
```

changes the owner from whomever the current owner of `mydoc.txt` to `bob`. Note that only the person(s) with write permissions to this file can perform this action.

Quick Tip: `chown user.group` will change the owner and group of the file

Using the `chgrp` command to set the group owner of files

As you saw in the previous hint, `user.group` will quickly change the group of a file or directory, and I personally prefer using `chown user.group`, but the original and proper way to change the group ownership of a file or a directory, can be done with the `chgrp` command, as follows:

```
chgrp users mydoc.txt
```

...which changes the group ownership of `mydoc.txt` to `users`.

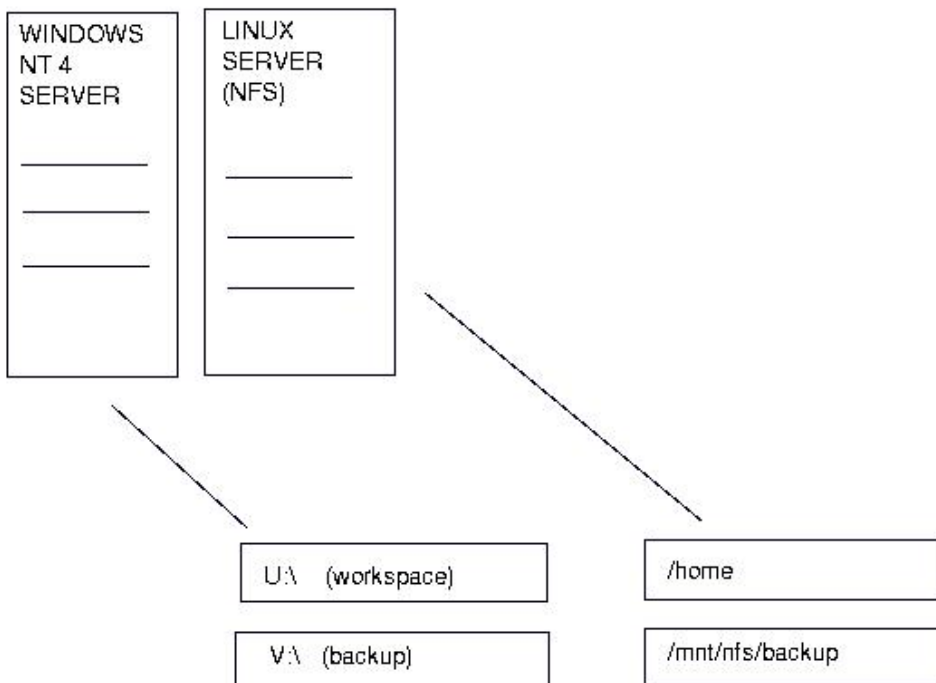
Even further in the depths of network file management: NFS

Some of you will have a brief idea of what NFS is, and some of you will have never before heard of NFS, and because of this, I will describe NFS from the top. If you know completely what it is, and just want to know how to set it up, feel free to scroll down a bit until we get to the section 'Using NFS'.

What Exactly *is* NFS?

NFS is a method that you can easily and very efficiently share files across a network in a Unix environment. Almost all types of Unix support NFS, and although not natively, Windows NT can be told to read NFS volumes.

A distinct advantage that NFS takes over the 'Windows Shares' mechanism (the system used to share files in Windows), is that NFS does not require to assign a separate drive on your computer for the use of shared files. Imagine the following scenario:



NFS file sharing, side by side Windows File Sharing (CIFS / SMB).

The image shows that directories of files or drives themselves, under Windows, must be exported to a drive letter (of which only a possible 24 are available). With Linux and Unix, you can export any directory or drive, and it can be viewed under any name on the client.

Imagine the scenario: You have the directory `/shared/home` on your server (let's call it *alpha*). You want your other client (desktop) machines, *bravo* and *charlie* to be able to see this folder, and everything inside it. Your two client systems are to have their home directories inside this share. You use NFS for this purpose. Perceive the users mary (In this example, Mary will use the Linux PC called bravo), and bob (uses the desktop charlie). Mary's current login directory on bravo is `/home/mary`, and bob's login directory on charlie is `/home/bob`. Both of these

directories are local to bravo or charlie and cannot be seen on other machines. You make two folders inside /shared/home on Alpha, one called mary and one called bob.

You move everything inside mary on her computer (bravo) into her folder on alpha, and you do the same for bob.

You remove the /home directory and all its contents on bravo and charlie -- as it's now redundant.

You then export the directory /shared/home to bravo and charlie (and call it /home)

The net result of all of this is that mary and bob (to them anyway, nothing has changed), but to the administrator of the network on alpha, all of the files are stored on the alpha server, and any files they now request from their home directories is now taken from the folder /shared/home on alpha.

Using NFS

Taking the three servers, alpha, beta and charlie a little further, this is a demonstration of how you can actually set it up:

NFS works on the basis of one of the rpc (Remote Procedure Call) tools called **nfsd**. All good Linux distributions should have rpc tools already with them (although they may be disabled in the sysv init startup), make sure this is installed and ready to go, and make sure that you have installed nfsd on the system that you wish to export your directories or drives (in our case, this is alpha).

Edit the file /etc/exports which is where all your exported volumes are listed. On alpha the format of the file would look like this:

```
/shared/home  bravo (rw,no_root_squash)  charlie (rw,no_root_squash)
```

This is what the above line means to you and I:

- 1) Take the directory /shared/home
- 2) Allow the computer **bravo** to read and write (rw) to it, and allow root to be treated as root over exported filesystems (no_root_squash)
- 3) Allow the computer **charlie** to read and write (rw) to it, and allow root to be treated as root over exported filesystems (no_root_squash)

Obviously, alpha must know, either by the file /etc/hosts or by using DNS / NIS how to resolve the hostnames bravo or charlie. If you don't think you can manage this at your current stage of Linux networking skills, then I recommend simply replacing the hostname (ie: bravo) with the IP address of that particular host (ie: 10.0.0.2).

You may or may not know about the infamous `/etc/fstab`, but this file holds the key for configuring bravo and charlie with the information they need to see the 'share' that alpha has now been set up to provide.

If you investigate `/etc/fstab`, you will notice that it has all the points of where all your drives are mounted, and this file is read by Linux when it starts up, to mount each drive. NFS Volumes are no different from physical disks, they too get added to this list, assuming that you want the volume to be mounted every time you use the computer.

Adding this line to the `/etc/fstab` on bravo and charlie would then kick in the mounting of the nfs volume on boot each time:

```
alpha:/shared/home    /home    rsize=8192,wsiz=8192
```

And in english:

- 1) Take the export or 'share' on the system alpha (`/shared/home`) (`alpha:/shared/home`).
- 2) Mount it on this computer (either bravo or charlie) as `/home` (ps: an empty directory with the name `/home` must exist to put this in). (`/home`)
- 3) Make the amount of retrieved bytes per block 8K / 8192 bytes (`rsize=8192`)
- 4) Make the amount of sent bytes per block 8K / 8192 (`wsiz=8192`)

Parts 3 and 4 of the statement are not necessary, but they help to improve efficiency over the network.

Also note that again, if you haven't set up an `/etc/hosts.txt` or DNS to resolve the name alpha, use its IP address in the place of the '**alpha**:', to read something like '**10.0.0.1**:'.

Once you are sure that `nfsd` is running correctly, you can do this by typing `ps ax | grep "nfsd"` at the shell. If you get a process called `nfsd` appearing (don't confuse this with `grep nfsd`, which is what you just typed), then you know that `nfsd` is running, and it sounds like it's configured correctly.

Now all you need to do is either start `nfs` manually on alpha, or reboot it, so that the startup script brings it up. You could reboot bravo and charlie, but preferably, just typing `'mount -a'` at the root prompt on both of the systems is enough for Linux to re-read its `/etc/fstab` file and cotton on to the fact that it has a new entry. If you get any errors when it re-reads this configuration, make sure to double check all of your settings.

Further Reading

[The Linux Documentation Project](#) should have the definitive guide to using NFS,

although it's possibly not too newbie-friendly.

Typing `man nfsd`, `man exports` and `man fstab` will all help you, and mainly `man exports`.

How to Mount Windows or Samba Shares Permanently

This howto describes how to mount Windows CIFS (SMB) shares permanently. The shares might be hosted on a Windows computer/server, or on a Linux/UNIX server running [Samba](#). This document also applies to SMBFS shares, which are similar to CIFS but are deprecated and should be avoided if possible ([link](#)).

This attribution is based on the original Ubuntu document ([link](#)), written by Contributors to the Ubuntu documentation wiki.

(This document does *not* describe how to host the shares yourself, only how to access shares that are hosted somewhere else. For hosting shares, use [Samba](#).)

We're assuming that:

- Network connections have been configured properly.
- Your local username is *ubuntuusername*.
- Share username on Windows computer is *msusername*.
- Share password on Windows computer is *msspassword*.
- The Windows computer's name is *servername* (this can be either an IP address or an assigned name).
- The name of the share is *sharename*.
- You want to mount the share in */media/windowsshare*.

```
sudo apt-get install cifs-utils
```

On older systems:

```
sudo apt-get install smbfs
```

First, let's create the mount directory. You will need a separate directory for each mount.

```
sudo mkdir /media/windowsshare
```

Then edit your `/etc/fstab` file (with root privileges) to add this line:

```
//servername/sharename /media/windowsshare cifs  
guest,uid=1000,icharset=utf8 0 0
```

Where;

- **guest** indicates you don't need a password to access the share,
- **uid=1000** makes the Linux user specified by the id the owner of the mounted share, allowing them to rename files,
- **icharset=utf8** allows access to files with names in non-English languages. This doesn't work with shares of devices like the Buffalo Tera Station, or Windows machines that export their shares using ISO8895-15.
- If there is any **space in the server path**, you need to replace it by `\040`, for example `//servername/My\040Documents`

After you add the entry to `/etc/fstab` type:

```
sudo mount -a
```

This will (re)mount all entries listed in `/etc/fstab`.

The quickest way to auto-mounting a password-protected share is to edit `/etc/fstab` (with root privileges), to add this line:

```
//servername/sharename /media/windowsshare cifs
username=msusername,password=mppassword,icharset=utf8,sec=
ntlm 0 0
```

This is *not* a good idea however: `/etc/fstab` is readable by everyone and so is your Windows password in it. The way around this is to use a credentials file. This is a file that contains just the username and password.

Using a text editor, create a file for your remote servers logon credential:

```
gedit ~/.smbcredentials
```

Enter your Windows username and password in the file:

```
username=msusername
password=mppassword
```

Save the file, exit the editor.

Change the permissions of the file to prevent unwanted access to your credentials:

```
chmod 600 ~/.smbcredentials
```

Then edit your `/etc/fstab` file (with root privileges) to add this line (replacing the insecure line in the example above, if you added it):

```
//servername/sharename /media/windowsshare cifs
credentials=/home/ubuntuusername/.smbcredentials,icharset=
utf8,sec=ntlm 0 0
```

Save the file, exit the editor.

Finally, test the `fstab` entry by issuing:

```
sudo mount -a
```

If there are no errors, you should test how it works after a reboot. Your remote share should mount automatically.

Special permissions

If you need special permission (like `chmod` etc.), you'll need to add a *uid* (short for 'user id') or *gid* (for 'group id') parameter to the share's mount options.

```
//servername/sharename /media/windowsshare cifs
uid=ubuntuuser,credentials=/home/ubuntuuser/.smbcredentials
,iocharset=utf8,sec=ntlm 0 0
```

Login errors

If you get the error "mount error(13) permission denied", then the server denied your access. Here are the first things to check:

- Are you using a valid username and password? Does that account really have access to this folder?
- Do you have whitespace in your credentials file? It should be `password=mspassword`, not `password = mspassword`.
- Do you need a domain? For example, if you are told that your username is SALES\sally, then actually your username is sally and your domain is SALES. The `fstab` entry should read: `...username=sally,password=pass,domain=SALES,...`
Or: `...credentials=/path/to/file,domain=SALES,...`
- Is the security setting correct? The most common is `sec=ntlm`, but you can also try the other options listed at the [mount.cifs man page](#). The man page list leaves out the option `sec=lanman` for some reason, but you should try that one as well (see [discussion](#)).

Unprotected network folder won't automount

I've had a situation where an unprotected network folder wouldn't automount during bootup, but after manually entering "sudo mount -a" was mounted correctly. I solved this by replacing the "guest" option by "username=guest,password=". If anyone has an explanation for this, please leave a comment.

```
//servername/sharename /media/windowsshare smbfs
username=guest,password=,uid=1000,iocharset=utf8,codepage=u
```

```
nicode,unicode 0 0
```

Mount during login instead of boot

If for some reason `/etc/rc0.d/S31umountnfs.sh` (networking problems for example) the automatic mounting during boot doesn't work, you can add the "noauto" parameter to your smbfs fstab entry and then have the share mounted at login.

In `/etc/fstab`:

```
//servername/sharename /media/windowsshare cifs  
noauto,credentials=/home/ubuntuusername/.smbpasswd 0 0
```

In `/etc/rc.local`:

```
mount /media/windowsshare  
exit 0
```

Slow shutdown due to a CIFS/Network Manager bug

If you use Network Manager, and are getting really slow shutdowns, it's probably because NM shuts down before unmounting the network shares. That will cause CIFS to hang and wait for 60 seconds or so. Here's how to fix it: `/etc/rc0.d/S31umountnfs.sh`

```
sudo ln -s /etc/init.d/umountnfs.sh  
/etc/rc0.d/K14umountnfs.sh  
sudo ln -s /etc/init.d/umountnfs.sh  
/etc/rc6.d/K14umountnfs.sh
```

Ubuntu 12.04 already runs **umountnfs.sh** at reboot and shutdown by default (`/etc/rc0.d/S31umountnfs.sh` and `/etc/rc6.d/S31umountnfs.sh`) so this is no longer necessary.

CIFS Options Deprecated

20 Feb 2008 TW

Using `dmask` or `fmask` in the `fstab` file produces the following warnings:

WARNING: CIFS mount option 'dmask' is deprecated. Use 'dir_mode' instead.

WARNING: CIFS mount option 'fmask' is deprecated. Use 'file_mode' instead.

Instead use this format: `file_mode=0777,dir_mode=0777` . Or in some cases you might need to use `file_mode=0777,dir_mode=0777,nounix` ([see discussion](#))

Use of tilde in pathnames such as "credentials=~/.smbcredentials"

20 Feb 2008 TW

Curiously, using `credentials=~/.smbcredentials` in `fstab` didn't work. I had to use the full path, i.e. `/home/username/.smbcredentials`

(This is likely because the tilde `"~"` is only a shell short-hand alias for `"$HOME"`; it isn't something recognized system-wide by all programs, especially not in a system file table where the concept of `"HOME"` doesn't really exist. -lan!)

Historic Items (older versions of Ubuntu)

Mount password protected shares using `libpam_mount` (Ubuntu 9.04)

In addition to the initial assumptions, we're assuming that

- **Your username and password are the same on the Ubuntu machine and on the network drive.**

Install `libpam-mount`:

```
sudo apt-get install libpam-mount
```

Edit `/etc/security/pam_mount.conf.xml` using your preferred text editor.

```
gksudo gedit /etc/security/pam_mount.conf.xml
```

First, we're moving the user specific config bits to a file which users can actually edit themselves: remove the commenting tags (<!-- and -->) surrounding the section called <userconf name=".pam_mount.conf.xml" />. Save the file when done. With this in place, users can create their own ~/.pam_mount.conf.xml.

```
gedit ~/.pam_mount.conf.xml
```

Add the following:

```
<?xml version="1.0" encoding="utf-8" ?>

<pam_mount>

<volume options="uid=%(USER),gid=100,dmask=0700" user="*"
mountpoint="/media/windowsshare" path="sharename"
server="servername" fstype="cifs" />

</pam_mount>
```

The material on this post is available under a free license, see [Copyright / License](#) for details.

Disable logging in with the root account

Say you got a new server set up on a VPS system, such as DigitalOcean. These servers often are set up with a root account and not a non-privileged account by default. We all know this is a bad thing, so once you have created a user that you'd like to make the admin user, who has sudo access, you can disable the ability to log in as root by issuing:

```
sudo passwd -l root
```

Note that this does not disable the account, merely disables logging in directly. You can always become root by issuing `sudo -s` to obtain a root shell :)

Quick tip: Add a user to the sudoers group

Situation: A new user on your server needs to have admin, or 'root' user access. Rather than enabling the root account, or giving out the password details to your elevated account, add the new user to the sudoers group, and hey presto - they will get sudo access with their own account and password:

```
usermod -aG sudo  
<username_of_person_you_want_to_give_root_to>
```

Adding users to groups

If you have a look at the `/etc/groups` file, you'll see a list of all the user groups on your Linux server (or desktop).

But first off..

What are groups?

Let's imagine that you are the user 'testuser'. By default, when you add a user on a Linux system, unless you choose otherwise when you create it, the user will be created in a group of his/her own. This group would also be called 'testuser'. This group contains exactly one user: testuser. Great.

However, groups allow you and other users of the same system to share permissions: access to documents, programs and more. For example, if you testuser was suddenly added to a group called 'admins', and that admins group had access to many files in otherwise confidential areas of the Linux system (for example, configuration files in the `/etc` folder), then 'testuser' would now also be able to access these files also.

Here's an excerpt from a typical directory layout (using the command `ls -l`):

```
-rw-r----- 1 syslog adm 5740 Jul 4 23:06 auth.log
```

The above file belongs to the user 'syslog', and also the group called 'adm'. The permissions of the file allow the user 'syslog' to read and write to the file (rw-), the group 'adm' can read the file only (r--), and everyone else, can't do anything to the file (---). If we added our fictitious users 'testuser' to the group 'adm', then they would be able to read this file.

How do I create a New Group?

To add a new group, all you need to do is use the `groupadd` command like so:

```
groupadd <groupname>
```

Add an Existing User to a Group

Next we'll add the user 'testuser' to the group 'adm':

```
usermod -a -G adm testuser
```

Change a User's Primary Group

If you want to switch the primary group that a user is assigned to, use the `usermod` command with the lower case `g` switch:

```
usermod -g <groupname> username
```

View the Group Assignments of a User

If you want to view which groups a user is a member of, you can use the `id` command. It shows you the `uid` (user ID number), the `username`, the `gid` (the group ID number), the `group name`, and the `gid` and `group names` of any additional groups the user may be part of. You may also specify someone else's username to view their details.

```
$ id

uid=1000(testuser) gid=1000(testuser)
groups=1000(testuser), 4(adm)
```

How to look at all the groups on a system and edit specific details (advanced).

If you are willing to get your hands dirty (and yes, this means you could severely break things), then you can run the `vigr` command, which allows you to edit the `groups` file (you need to be root).

```
$ vigr
```

you can manually change the `group names`, `gids` and `user memberships` of any groups within the text editor.

Add a New User and Assign a Group in One

Command

Sometimes you might need to add a new user that has access to a particular resource or directory, like adding a new FTP user. You can do so with the `useradd` command, using the `-G` flag:

For example, to create a brand new user named 'testuser2' to the postfix group, you'd issue the following command:

```
useradd -G postfix testuser2
```

Don't forget to assign a password for that user:

```
passwd testuser2
```

Add a User to Multiple Groups

You can add a user to more than one group by specifying them in a comma-delimited list:

```
usermod -a -G postfix,adm,othergroup testuser2
```

Finally, Have a look at the `adduser` and `addgroup` commands also. `Adduser` makes it easy to interactively make new users without worrying about remembering the flags.

How To Set Up SSH Keys

About SSH Keys

SSH keys provide a more secure way of logging into a virtual private server with SSH than using a password alone. While a password can eventually be cracked with a brute force attack, SSH keys are nearly impossible to decipher by brute force alone. Generating a key pair provides you with two long string of characters: a public and a private key. You can place the public key on any server, and then unlock it by connecting to it with a client that already has the private key. When the two match up, the system unlocks without the need for a password. You can increase security even more by protecting the private key with a passphrase.

This guide was mainly written by the folks at [DigitalOcean](https://www.digitalocean.com). Check out their great, cost competitive server options over at [digitalocean.com](https://www.digitalocean.com)

Step One—Create the RSA Key Pair

The first step is to create the key pair on the client machine (there is a good chance that this will just be your computer):

```
ssh-keygen -t rsa
```

Step Two—Store the Keys and Passphrase

Once you have entered the Gen Key command, you will get a few more questions:

```
Enter file in which to save the key  
(/home/demo/.ssh/id_rsa):
```

You can press enter here, saving the file to the user home (in this case, my example user is called demo).

```
Enter passphrase (empty for no passphrase):
```

It's up to you whether you want to use a passphrase. Entering a passphrase does have its benefits: the security of a key, no matter how encrypted, still depends on the fact that it is not visible to anyone else. Should a passphrase-protected private key fall into an unauthorized users possession, they will be unable to log in to its associated accounts until they figure out the passphrase, buying the hacked user some extra time. The only downside, of course, to having a passphrase, is then having to type it in each time you use the Key Pair.

The entire key generation process looks like this:

```
ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key
(/home/demo/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in
/home/demo/.ssh/id_rsa.
Your public key has been saved in
/home/demo/.ssh/id_rsa.pub.
The key fingerprint is:
4a:dd:0a:c6:35:4e:3f:ed:27:38:8c:74:44:4d:93:67 demo@a
The key's randomart image is:
+-----+
|           .oo.   |
|          .  o.E  |
|         + .  o   |
|        . = = .   |
|         = S = .   |
|       o + = +    |
|        . o + o .  |
|           . o    |
|                   |
+-----+
```

The public key is now located in /home/demo/.ssh/id_rsa.pub The private key (identification) is now located in /home/demo/.ssh/id_rsa

Step Three—Copy the Public Key

Once the key pair is generated, it's time to place the public key on the virtual

server that we want to use.

You can copy the public key into the new machine's `authorized_keys` file with the `ssh-copy-id` command. Make sure to replace the example username and IP address below.

```
ssh-copy-id user@123.45.56.78
```

Alternatively, you can paste in the keys using SSH:

```
cat ~/.ssh/id_rsa.pub | ssh user@123.45.56.78 "mkdir -p  
~/.ssh && cat >> ~/.ssh/authorized_keys"
```

No matter which command you chose, you should see something like:

```
The authenticity of host '12.34.56.78 (12.34.56.78)' can't  
be established.  
RSA key fingerprint is  
b1:2d:33:67:ce:35:4d:5f:f3:a8:cd:c0:c4:48:86:12.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '12.34.56.78' (RSA) to the list  
of known hosts.  
user@12.34.56.78's password:  
Now try logging into the machine, with "ssh  
'user@12.34.56.78'", and check in:  
  
~/.ssh/authorized_keys  
  
to make sure we haven't added extra keys that you weren't  
expecting.
```

Now you can go ahead and log into `user@12.34.56.78` and you will not be prompted for a password. However, if you set a passphrase, you will be asked to enter the passphrase at that time (and whenever else you log in in the future).

Optional Step Four—Disable the Password for Root Login

Once you have copied your SSH keys unto your server and **ensured that you can log in with the SSH keys alone**, you can go ahead and restrict the root login to only be permitted via SSH keys.

In order to do this, open up the SSH config file:

```
sudo nano /etc/ssh/sshd_config
```

Within that file, find the line that includes `PermitRootLogin` and modify it to ensure that users can only connect with their SSH key:

```
PermitRootLogin without-password
```

Put the changes into effect:

```
service openssh-server restart
```

SysAdmin Tips: ViM Text Editor 101 Guide

The ViM text editor, Vi Improved, has been around for a very, very long time. However, it remains arguably the most popular text editor amongst Unix users young and old due to it's speed, ability to adapt to almost any purpose, and the fact that it will be found on any Unix/Linux box anywhere. You can't say that for nano or even emacs.

Suffice it to say, learning how to the ViM a text editor effectively will make your life better, even if at first it drives you crazy.

The following slide deck is a takeaway and print off guide to using ViM effectively, and starts you off from zero knowledge.

ViM 101 Essentials Guide



What is Docker (and Linux containers?)

A while ago, around the release of the Linux kernel version 3, the concept of namespaces and containerization was introduced through a module called lxc (Linux containers).

The idea behind a container is sort of similar to the idea of a virtual machine. For example, with virtualisation, you have a server (the 'host') running something like KVM or VMware. The machines running under it are called guests. They are fully self-contained computers, running on top of the host.

Containers take this concept to the next level. Containers help developers especially (but also systems administrators) deploy applications or services rapidly. Containers are very, small Linux machines that run as a normal Linux process (in *userspace*). An average Linux server is gigabytes in size, and has a kernel choc-full of handy drivers for all sorts of hardware and so forth. A container, conversely is only a few hundred megabytes in size. A VM will boot in minutes, a container will start up, including its intended application, in a few seconds or less. It runs just as you'd expect any normal Linux application to run - as a process you can see using the `ps` command.

The below diagram from the folks at [CoreOS](#) shows the relation between containers and the Operating System well. On the top, you can see a normal Linux server, running all of the usual services you might imagine, such as Java, the nginx web server etc, there will also likely be a few other applications running too.

In the Figure 2, you can see an example of an Operating System that is geared towards containerisation. It has only limited, minimal applications, such as the ssh server running. All of the other applications are now running within completely isolated containers.

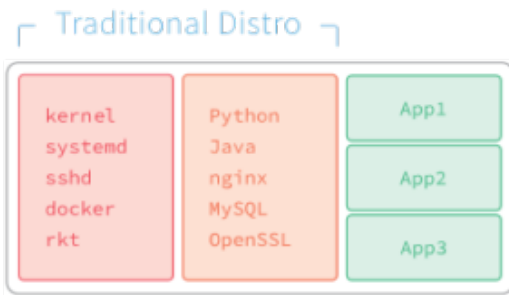


Figure 1: A traditional Linux server, running all of the usual services as well as the provided applications, all within the same stack, depending upon the server's shared software libraries.

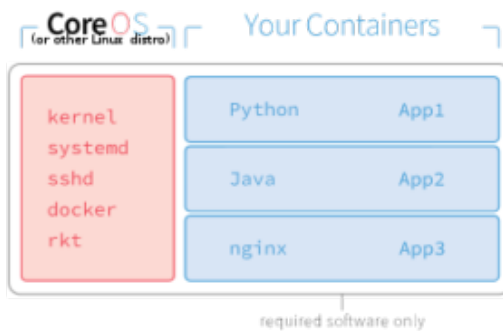


Figure 2: A minimal Linux distro, with all served applications running in discrete containers.

Ephemeral Beings

Containers are ideally made to be single purpose in nature: imagine you have an application that uses JBoss and PostgreSQL. You'd likely have a container for both JBoss and PostgreSQL components. The two containers are isolated from each other, but it is possible for them to be linked, so they can talk to each other. Containers are also designed to be ephemeral: once it's purpose has been served, you blow it away, you can always spin up another one in seconds. This also means that data stored inside a container is **not** persistent. If you want data, such as databases or web sites to remain, you can put them in an exported storage volume on the host server. In security best practice, you would update your base-image with any new patches or security hardening procedures, and then roll on your application. After this, you'd destroy your container from last week, and use the new, securer container the next week.

Containers fit great with the whole SaaS (software as a service model), and it's also enabling developers and operations staff to work together

in a friendlier way (read DevOps way). But let's not get ahead of ourselves here, tooling isn't going to fix cultural problems, that's a whole 'nother topic for another website!

Docker

Docker has been around for a while now. Docker took to Linux what lxc could not: ease of use. Once developers found out how empowering it was just to spin up a docker container on their laptop and be guaranteed it would operate in exactly the same way on a server somewhere else, with the greatest of ease, it quickly became a no-brainer.

Docker has its critics, some (like the [coreOS team](#)) believe that Docker isn't secure enough, and they are becoming too commercialised (that's why they made [rkt](#) as a competitor to Docker), but love it or hate it, Docker has a huge following, and not just in companies you'd expect, like Amazon and Google.

More Reading

You can read more about containerisation, over at the following websites:

[LinuxContainers.org](#)

[Docker: What is Docker?](#)

[Amazon AWS: What are Containers?](#)

[Rkt \(pronounced Rocket\), from CoreOS](#)

running a command against every line in a textfile

So, you have a text file like so:

```
file1  
file2  
..
```

And you want to run a command on each line of the file (say, `chmod 644`). Like all things UNIX, there are more than one way to do thing, but here's the quick and dirty answer:

```
xargs -0 -n 1 chmod 644 < <(tr \n \0 <filelist.txt)
```

So, this will run `xargs` on the std. input (`filelist.txt`).

`-n` is max arguments - in this case we have 1 arg. (the list of files in the file - per line).

`-0` means that input lines are terminated by a null char, not a whitespace, to keep things tidy.

Finally, we run `tr` which deletes characters, in this case removing the newlines and whitespace.

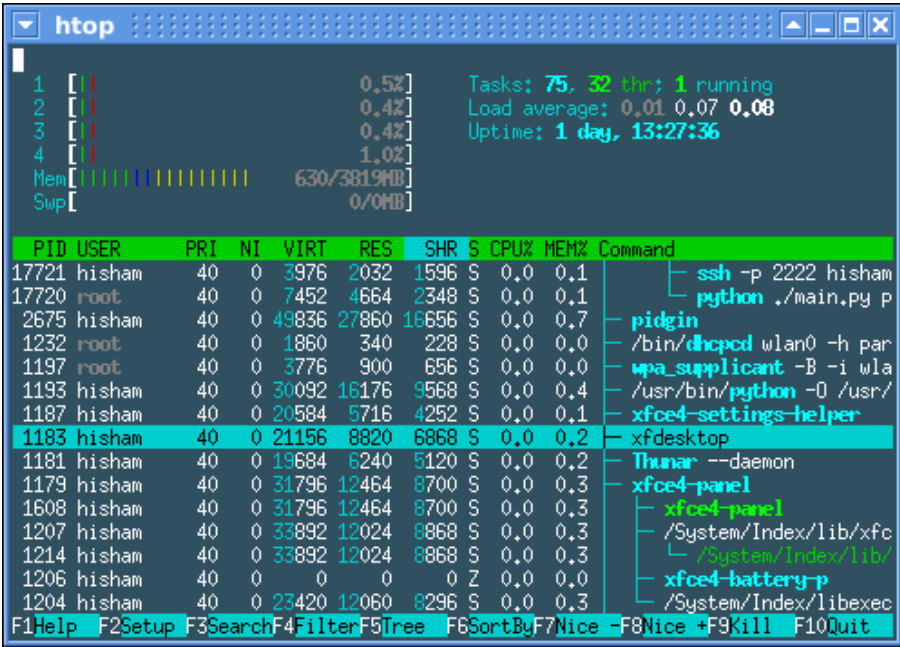
Monitoring network bandwidth, CPU and memory effectively

Here is a bunch of handy tips for today that will likely remain in your armoury forever.

As a Linux sysadmin it's sometimes difficult to visualise just what is causing a performance problem. Sure, it's easy enough to see which process is hogging the CPU with tools like 'top' or its fancier brother, htop. When it comes to figuring out the long term load on a machine or understanding how much memory and network bandwidth is being used can be a little more of a challenge if you aren't aware of the tools out there.

CPU & memory monitoring with (h)top

Use the F6 button in htop to sort by CPU or memory etc.



```
htop
 1 [ ] 0.5% Tasks: 75, 32 thr; 1 running
 2 [ ] 0.4% Load average: 0.01 0.07 0.08
 3 [ ] 0.4% Uptime: 1 day, 13:27:36
 4 [ ] 1.0%
Mem [|||||] 630/3819MB
Sup [ ] 0/0MB

 PID USER   PRI NI  VIRT  RES  SHR  S  CPU% MEM% Command
17721 hisham  40  0  3976 2032 1596 S  0.0  0.1 ssh -p 2222 hisham
17720 root     40  0  7452 4664 2348 S  0.0  0.1 python ./main.py p
2675  hisham  40  0  49836 27860 16656 S  0.0  0.7 pidgin
1232  root     40  0  1860 340 228 S  0.0  0.0 /bin/dhccpd wlan0 -h par
1197  root     40  0  3776 900 656 S  0.0  0.0 wpa_supplicant -B -i wla
1193  hisham  40  0  30092 16176 9568 S  0.0  0.4 /usr/bin/python -O /usr/
1187  hisham  40  0  20584 5716 4252 S  0.0  0.1 xfce4-settings-helper
1183  hisham  40  0  21156 8820 6868 S  0.0  0.2 xfdesktop
1181  hisham  40  0  19684 6240 5120 S  0.0  0.2 Thunar --daemon
1179  hisham  40  0  31796 12464 8700 S  0.0  0.3 xfce4-panel
1608  hisham  40  0  31796 12464 8700 S  0.0  0.3 xfce4-panel
1207  hisham  40  0  33892 12024 8868 S  0.0  0.3 /System/Index/lib/xfc
1214  hisham  40  0  33892 12024 8868 S  0.0  0.3 /System/Index/lib/
1206  hisham  40  0  0 0 0 Z  0.0  0.0 xfce4-battery-p
1204  hisham  40  0  23420 12060 8296 S  0.0  0.3 /System/Index/libexec

F1)help F2)Setup F3)Search F4)Filter F5)Tree F6)SortBy F7)Nice F8)Nice +F9)Kill F10)Quit
```

htop real time cpu analyser

Analysing CPU, Memory and Disk I/O over a measured time

To analyse the average CPU, memory and disk I/O load over a measured amount of time, use the `vmstat` tool. It is ugly looking in comparison to `htop` but once you understand the display it can be highly effective in understanding what's going on with the system except network utilisation. Note as well that virtualised guest servers might not give the true CPU & I/O figures as these can vary dynamically based on the hypervisor settings.

Like `top`, `vmstat` is almost ubiquitous in availability for each Linux version. `Vmstat` normally takes two arguments: the sample time and the number of samples to measure. So for example running

```
vmstat 1 100
```

Will make a sample each second and will perform the sample 100 times. By default `vmstat` will show you the output of the CPU load, memory/swap and block I/O, when it runs its 100 samples, it will give you the averages over the time samples for, in this case 100 seconds. If you wish to run `vmstat` continuously use 0 as the sample number. More information on the syntax and output of `vmstat` is available [here](#) (or use `man vmstat`).

Finally as a text based alternative you can brew this function in your `.bashrc` or in a shell script, this will allow you to execute it at intervals using the `at` command or schedule with `cron` or perhaps combine it with another script to make further analysis over time.

```
memcpu() echo "--- Top 10 cpu eating process ---"; ps auxf | sort -nr -k 3 | head -10;  
echo "--- Top 10 memory eating process ---"; ps auxf | sort -nr -k 4 | head -10;
```

Analysing network utilisation quickly

Monitoring network utilisation is arguably as important as your CPU and memory. The amount of built in tools that do this vary between distributions. There is a multitude of tools you can install via yum or apt-get in the respective distributions. You can try ntop or nmon. Today we are going to look at nload. Although ntop touts itself as the 'top' command of networking, it's a web based tool which whilst good, isn't as simple to get going as nload. To execute nload simply run it without any arguments and it will output the load on the current network interface.

```
Options:
===== <-- (-) page 1/2 (+) --> =
Refresh interval (ms):          500
Show multiple devices:         [ ]
Max Incoming deflection (kBit/s): 1024
Max Outgoing deflection (kBit/s): 1024
Smoothness of average:         9

Device eth0 [10.1.1.2] (1/1):
=====
Incoming:

| . . | . . | | #. | .# . | . | # | #
## #| ## ## |# # #| ## ## ## # #| |# # . ## #
## ## ## ## ## .# ## ## ## ## # ## # ## # ## #
## ## ## ## ## ## ## . ## ## . ## ## #| ## ## |# Curr: 821.45 kBit/s
## ## ## ## ## ## ## # ## ## # ## # ## ## ## ## Avg: 510.59 kBit/s
## ## ## ## ## ## ## # ## ## # ## # ## ## ## ## Min: 0.00 kBit/s
##,##,##,##,##,##,##,## ##,##, ##,##,##,##,##,##,## Max: 919.31 kBit/s
#####|##### Ttl: 16.64 MByte

Outgoing:

. . | . . | | #. | .# . | . | # | #
## #. ## ## .# | #. ## |# ## # |. .# # ## #
## ## ## ## ## # ## ## ## # ## # ## # #| ## #
## ## ## ## ## |# ## ## ## ## # ## #. ## #. #
## ## ## ## ## ## #| ## ## | ## # ## ## ## ## ## Curr: 821.45 kBit/s
## ## ## ## ## ## ## # ## ## # ## # ## ## ## ## Avg: 510.62 kBit/s
## ## ## ## ## ## ## # ## ## # ## | #. ## ## ## ## ## Min: 0.00 kBit/s
##,##,##,##,##,##,##,## ##|##| ##,##,##,##,##,##,##,## Max: 919.31 kBit/s
#####|##### Ttl: 16.22 MByte
```

nload at work

Nload does just what it says on the tin. The historic analysis makes it easy to see how busy the network is, unfortunately that won't show you what application is causing the load but there are apps which can help there too like the excellent [nethogs app](#). It looks and works just like top- showing processes by name and sorted by order of which process is chewing the most bandwidth.

```
NetHogs version 0.5.0, running at eth0
```

PID	USER	PROGRAM	SENT	RECEIVED
16651	tom	proftpd: tom - catv6062,exter	71,264	1,773 KB/sec
16652	tom	proftpd: tom - catv6062,exter	72,743	1,759 KB/sec
14106	tom	dcgui-qt	0,080	0,133 KB/sec
21423	miachel	psync	0,013	0,034 KB/sec
29578	arnouten	irssi	0,029	0,022 KB/sec

nethogs at work

In conclusion and further options

What I've demonstrated here are some great, quick analysis tools to get you out of a potentially difficult to diagnose issue. If you need longer term analysis of almost any aspect measurable then you should look to something like [nagios](#) and [combine it with rrdtool](#) to graph historical trend analysis. Look at [cacti](#) (rrd graphing) and [munin](#) (sort of like nagios + rrdtool + cacti in one easy package).

How to tar (compress) files up, excluding certain files or directories

If you've ever been making a backup of an entire Linux system, or maybe just a number of folders but there were certain folders or files that you didn't want to have in the backup or zip file, then Look no further than this Quick Tip!

First, change to the folder you want to zip, or back up and make sure you have permissions to access all of the files within the folder. For example, if the folder is / (root) then you will need superuser permissions so don't forget to run tar with the sudo command!

```
cd /folder_to_backup<br />
```

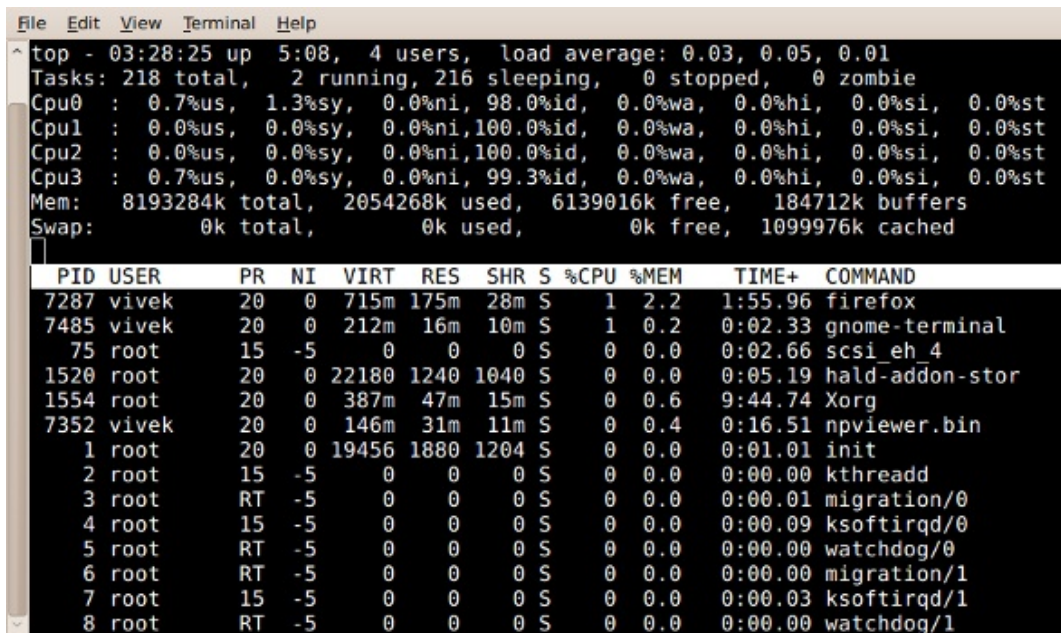
Next, you want to run the tar command to create the archive/zip file. The usual z (gzip compression), c (create), v (verbose), f (file) flags are used, but note that they come at the latter part of the command line. This placement seems important across differing distributions of Linux.

You can see that using the --exclude option we can specify the folders and/or files to exclude, you can have as many --exclude options as you need. Note how the path is prefixed with a period (denoting the current directory). This is important because the exclude flag matches text patterns, not actual filenames, and the pattern starts with ./ - You can also use other regular expressions. For example you can use a wildcard such as file* to match any file or folder name beginning with the word *file*.

```
tar --exclude='./folder_to_exclude' --  
exclude='./myfolder/file.txt' -zcvf /backup/filename.tgz .
```

Analysing system performance with 'Top'

There are literally hundreds of guides on the Internet detailing how to use the 'top' command. A very handy command-line tool that has come with [UNIX](#) since back in the dark ages, however not all of these guides are directed flatly at the [new Linux user](#). This one won't go into loads of detail but will give you the basics.



```
File Edit View Terminal Help
^ top - 03:28:25 up 5:08, 4 users, load average: 0.03, 0.05, 0.01
Tasks: 218 total, 2 running, 216 sleeping, 0 stopped, 0 zombie
Cpu0  : 0.7%us, 1.3%sy, 0.0%ni, 98.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu1  : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu2  : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu3  : 0.7%us, 0.0%sy, 0.0%ni, 99.3%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem:   8193284k total, 2054268k used, 6139016k free, 184712k buffers
Swap:  0k total,    0k used,    0k free, 1099976k cached

  PID USER   PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 7287 vivek   20   0  715m 175m 28m  S   1    2.2   1:55.96  firefox
 7485 vivek   20   0  212m  16m 10m  S   1    0.2    0:02.33  gnome-terminal
   75 root    15  -5    0    0    0  S   0    0.0    0:02.66  scsi_eh_4
1520 root    20   0 22180 1240 1040  S   0    0.0    0:05.19  hald-addon-stor
1554 root    20   0  387m  47m 15m  S   0    0.6    9:44.74  Xorg
 7352 vivek   20   0  146m  31m 11m  S   0    0.4    0:16.51  npviewer.bin
    1 root    20   0 19456 1880 1204  S   0    0.0    0:01.01  init
    2 root    15  -5    0    0    0  S   0    0.0    0:00.00  kthreadd
    3 root    RT  -5    0    0    0  S   0    0.0    0:00.01  migration/0
    4 root    15  -5    0    0    0  S   0    0.0    0:00.09  ksoftirqd/0
    5 root    RT  -5    0    0    0  S   0    0.0    0:00.00  watchdog/0
    6 root    RT  -5    0    0    0  S   0    0.0    0:00.00  migration/1
    7 root    15  -5    0    0    0  S   0    0.0    0:00.03  ksoftirqd/1
    8 root    RT  -5    0    0    0  S   0    0.0    0:00.00  watchdog/1
```

The top command in action

Why would I want to run 'top'?

Top is a great utility to find out if your [Linux machine](#) is running slowly, or perhaps you want to see what a server is doing most of the time, top tells you loads of things about how well your box is performing and can be compared to tools like the [Windows Task Manager](#).

How do I run 'top'?

Top is a [command-line](#) tool. That is, you need to run the Terminal or Konsole

program in order to run it. For example, in [Ubuntu](#), click on Applications, then click Accessories, then click 'Terminal'. You will be presented with a [command prompt](#). Type the word **top** (in lower case) and press return. You will then see the 'top' program running.

What am I seeing here?

With any luck, your terminal window should look a bit like this:

```
top - 16:17:41 up 100 days, 18:01, 4 users, load average: 0.20, 1.13, 1.77
Tasks: 126 total, 1 running, 125 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.1%sy, 0.0%ni, 99.8%id, 0.2%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 16383952k total, 15630644k used, 753308k free, 4180008k buffers
Swap: 7815580k total, 64k used, 7815516k free, 10127600k cached
```

```
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
1 root 20 0 1948 600 508 S 0 0.0 0:21.74 init
```

Whoa! What is all that nonsense?! Have no fear, it will all make sense in a second and you'll be able to impress all your friends with your new found knowledge!

The first line shows all of the stats that the command 'uptime' shows. For example, you can see the time of the system, how many days the system has been running (the 'uptime') - in this case my system has been running for 100 days, 18 hours, 1 minute. There are four active users on the system and finally you see the 'load average' figures.

The load average shows how many processes (program tasks) are ready to run over three time averages: 1 minute, 5 minutes and 15 minutes. You can tell that the 'load' on this box has come down from 1.77 to 0.20 in the last 15 minutes. Typically, you will find that a load average of over **10** is fairly high and you will definitely start to notice the computer being slower.

The next line is useful, but not as useful as the last one. The amount of tasks currently waiting and running is listed, note that 125 out of the 126 tasks at this time were 'sleeping'. The only running process was in fact 'top', everything else was doing nothing - just waiting around for something to happen, thus they are sleeping. You'll also notice that there are 0 zombie processes - these are when a process spawns (starts) another process (eg a [child process](#)) and the parent process fails and leaves a child process behind. The processes are still running but have nothing to do and nothing to speak to, so essentially they are 'Zombified processes!'. They are often difficult to get rid of, but I rarely see them these days

unless you aren't looking after your system.

The next line will show you how much % of the CPU is being used, and in what states. If the CPU is 100% used for a blink and then back to around 5% use, this is quite normal and you will note that it happens quite a lot. The %us means this is how much percent of the system CPU usage is being occupied by user tasks (eg a task that you run as user 'bob'. The opposite of root or [system processes](#)). The %sy is the amount of system processes are using the CPU. Next, %ni means the amount of processes in percent that are 'niced' processes, eg processes that have had their normal weighting of priority adjusted in some way. Finally the %id is the percentage of the CPU that is currently idle, waiting for instructions, you can see that this box is really doing very little here, thus the high idle value.

There is a lot to say about Mem (Memory) and Swap ([Virtual Memory](#)) usage, beyond the scope of this article, but needless to say, you should always expect the amount of free memory to be low - this is by design, it's not like the old days in Windows or DOS. [Linux](#) automatically allocates most of your available RAM memory to use in caches.

The rest of the top program shows you the 'top' running processes (thus why the program is called top!). By default, it shows you the top processes sorted by CPU usage.

The example below shows an idle system, but you might see that a number of processes are above the process '1' (called init), and these are all chewing up more CPU usage. Here is what all the fields mean in that bar along the top:

PID - Process ID. The unique number given to each process on the system. The [init process](#) always has PID 1 because it is the first thing that runs, and it spawns all other processes. Don't kill this process unless you want to reboot your box!

USER - Username. This is the user or username that 'owns' the process in question. This way you can quite quickly see which user or users are chewing up most of the system's utilisation. Remember that the 'root' user is the system user.

PR - Priority. This is the priority of the process. This is often of little use to you as the kernel automatically works out the priority of a process depending upon the load and usage of a process. The higher the number, the lower the priority, +20 being the lowest priority, -20 the highest.

NI - 'Nice' Value. The 'nice' value can be between -19 and +20. This over-rides the

priority of a process a bit, so if you have a big heavy duty process that you want to run, but don't want it to overpower everything else, you can 'renice' a priority to +20. If you want it to go above the normal threshold of kernel prioritisation (0), then you need to be the root user, you can renice a process down to -19 to beef up the priority of a task over any others.

VIRT - Virtual Memory allocated. This is the amount of virtual memory the process is using presently.

RES - Resident Memory allocated. This is the amount of 'real' memory allocated.

SHR - Shared Memory allocated. Processes can share memory with other processes, this is the amount of memory they are using which is considered to be 'shared' memory.

S - Status. This is the status of the process, it will either be R (Running), S (Sleeping) or Z (Zombie).

%CPU - This is the amount in percent that this process is using of the CPU at that instance.

%MEM - This is the amount of allocated memory that this process is using at that instance. Often you will find that this is fairly low.

TIME+ - This is the amount of **CPU time** a process is using in hundredths of a second.

COMMAND - This is the actual command that is running, or the name of the process.

Cool, what else can I do with top?

There are a few keys you can press within top, that will help you analyse other parts of your system's performance.

Kill - If you press k and enter a process ID (PID), you will kill (close down) that process. Be careful with this though, as if you kill a process that you shouldn't, the system can become unstable, especially if you are running top as the super-user (root). If you are asked for a 'signal' to give a process, there are a number of signals you can give but 15 and 9 are the most common. See 'man kill' for an idea of what each one does. Essentially 15 will terminate a normal process gracefully, 9 will kill it

straight away (not graceful - doesn't have time to save it's state or data).

Quit - to quit out of top, press q (lower case q).

Renice - to renice a process (see above section) use the r key.

Sort by memory usage - press lower case m.

Sort by CPU Usage - Press capital P. This is the default view

Further Usage and Reading

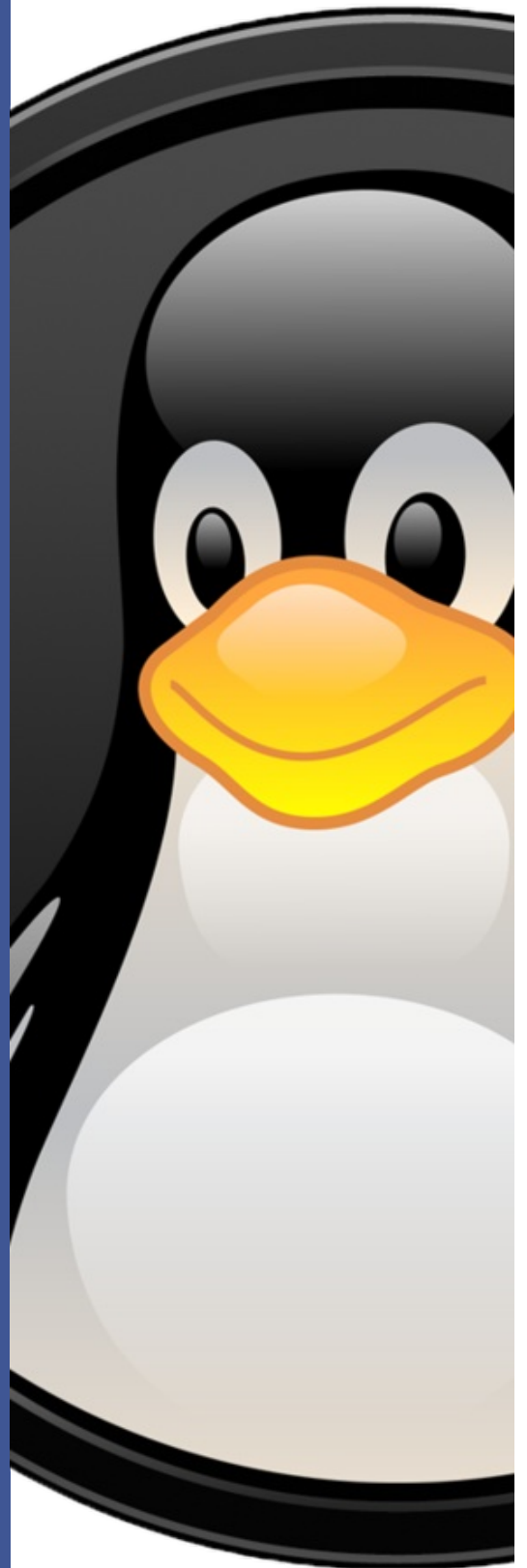
For further information, at the [command line](#), type **man top** and you will see the manual page on the top program which gives you detailed information for top. You'll also find plenty of other guides on the Internet that go into further depth, but hopefully this helps you to diagnose your system's performance. For example, if something is running slowly (often a problem with programs like firefox crashing and chewing up CPU usage). You are likely to see firefox at the top of the top list. You can kill it by pressing k and entering the PID number of firefox, then pressing return. Once that's done, unless there are other processes still chewing up the CPU, you should notice things returning to normal.

Finally, there's also a newer, prettier (arguably nicer) version of top, which although is not on every linux system, it can be easily installed. It's called [htop](#).

Thanks for Reading!

Don't forget to check out the website for all the latest updates, quick tips, video tutorials. Support the author in keeping it free, too!

linuxnewbieguide.org



made with
Beacon